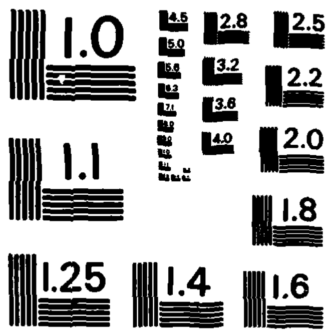
AD-A121 958 SUPPLY POINT LOGISTICS INTEGRATED COMMUNICATIONS ENVIRONMENT (SPLICE) LOC.. (U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA K A INMAN ET AL. JUN 82 1/2

UNCLASSIFIED F/G 17/2.1 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

②

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

SUPPLY POINT LOGISTICS INTEGRATED COMMUNICATIONS ENVIRONMENT (SPLICE) LOCAL AREA COMPUTER NETWORK DESIGN ISSUES FOR COMMUNICATIONS

by

Kenneth A. Inman, Jr.

and

Robert C. Marthouse, Jr.

June 1982

Thesis Advisor: N. F. Schneidewind

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A121 958	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Supply Point Logistics Integrated Communications Environment (SPLICE) Local Area Computer Network Design Issues for Communications		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis June 1982
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Kenneth A. Inman, Jr. Robert C. Marthouse, Jr.		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE June 1982
		13. NUMBER OF PAGES 161
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report)
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) SPLICE; Supply Point Logistics Integrated Communications Environment; Local Computer Network; Network Protocol; End-to-End Protocol.		

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This thesis examines the topology and transmission mediums for a local computer network to support interconnection of heterogenous computing devices within the Supply Point Logistics Communications Environment (SPLICE). A topology and appropriate network protocols for management of the intranetwork communications are recommended. Additionally, a protocol to ensure proper delivery of messages which must pass outside the local network to reach another SPLICE configuration via an interconnecting network is discussed.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Supply Point Logistics Integrated Communications Environment (SPLICE) Local Area Computer Network Design Issues for Communications

by

Kenneth A. Inman, Jr.
Captain, United States Marine Corps
B.S., Oklahoma University, 1977

and

Robert C. Marthouse, Jr.
Lieutenant, Medical Service Corps, United States Navy
B.S., George Washington University, 1977

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
June 1982

Authors: [illegible]

[illegible]

Approved by: [illegible]
Thesis Advisor

[illegible]
Second Reader

[illegible]
Chairman, Department of Administrative Sciences

[illegible]
Dean of Information and Policy Sciences

ABSTRACT

This thesis examines the topology and transmission mediums for a local computer network to support interconnection of heterogenous computing devices within the Supply Point Logistics Communications Environment (SPLICE). A topology and appropriate network protocols for management of the intranetwork communications are recommended. Additionally, a protocol to ensure proper delivery of messages which must pass outside the local network to reach another SPLICE configuration via an interconnecting network is discussed.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

Local area computer networks provide for the interconnection of data processing and computing devices located within a limited geographical area. Because of the present and probable future increases in the number of such devices at the Navy's supply points and inventory control points, networking of the devices within the local area offers the potential to efficiently share available resources. Integration of all organizational computer resources is the goal of the Supply Point Logistics Integrated Communications Environment (SPLICE) [Ref. 1]. Implementation of local area computer networks at all stockpoints (SP's) and inventory control points is a means to this system integration. Additionally, a local network structure capable of providing compatible interconnection of various terminals, data processing devices, word processors, gateways to other computer networks, and of virtually any type of digital communication device, can provide an extremely flexible, highly reliable, environment for the future evolution of SPLICE configurations.

A major advantage of local area networks which is particularly important for SPLICE is that, once implemented, the local network can support virtually any type of system transition. For example, as any particular host computer on the network becomes obsolete, a new machine can be attached to the network. During the transition period users can access processes in both machines.

Another significant aspect of a well-designed local network is that it can support a long term, vendor independent, expansion strategy.

With appropriate design, the network can support communications among heterogenous devices. Consequently, when a new vendor's machine is interfaced to the network, little modification or enhancement of old or existing network hosts would be required.

Local area networks (LAN's) have been installed and implemented in various forms, for various functions and with varying success [Ref. 2, 3, 4, 5]. The functions of a local computer network are primarily aimed at providing a communications means for processes resident in the hosts which are connected to the network. Certain basic parameters which govern the character of the network are its topology, physical transmission medium, and protocols for network management. The purpose of this thesis is to examine these design parameters, determine their functions, and when appropriate, recommend a particular design which will support a general SPLICE local area computer network. Additionally, the thesis will examine an appropriate end-to-end protocol, which will allow for interconnection of a SPLICE local area network to another SPLICE configuration, via some, as yet, undefined packet switched communications network.

A. THE OPEN SYSTEMS INTERCONNECTION MODEL

The International Standards Organization (ISO) and the American National Standards Institute (ANSI) have developed a standard reference model for interconnecting open systems into operational networks. This model is called the Open Systems Interconnection Model [Ref. 6]. Open systems are computer systems which are "open" to one another for the exchange of information. This ISO/ANSI Open Systems Interconnection

(OSI) model is depicted in Figure 1.1. The seven layers of the model are described as follows:

1. Application Layer (Layer 7)

The application layer provides the facilities required to support process selection, activation and synchronization in the open system application environment. This layer might consist of several user level protocols such as a remote batch processing protocol, a distributed data base synchronization protocol, a file transfer protocol or a terminal control protocol.

2. Presentation Layer (Layer 6)

The presentation layer provides required transformation of data being transferred between two processes. This might consist of such functions as encrypting/decrypting, compacting/expanding, or implementation of a virtual terminal protocol.

3. Session Layer (Layer 5)

The session layer provides "sessions" or cooperative relationships to support the communications between user processes. It basically binds cooperating process into temporary relationships to control any data exchanges between the processes.

4. Transport Layer (Layer 4)

The purpose of the transport layer is to provide reliable end-to-end transport of messages across any arbitrary configuration of inter-connected networks. This layer is responsible for reliable communications within a logical network regardless of the reliability of any intervening communications facilities or other networks which may be utilized for physical movement of data.

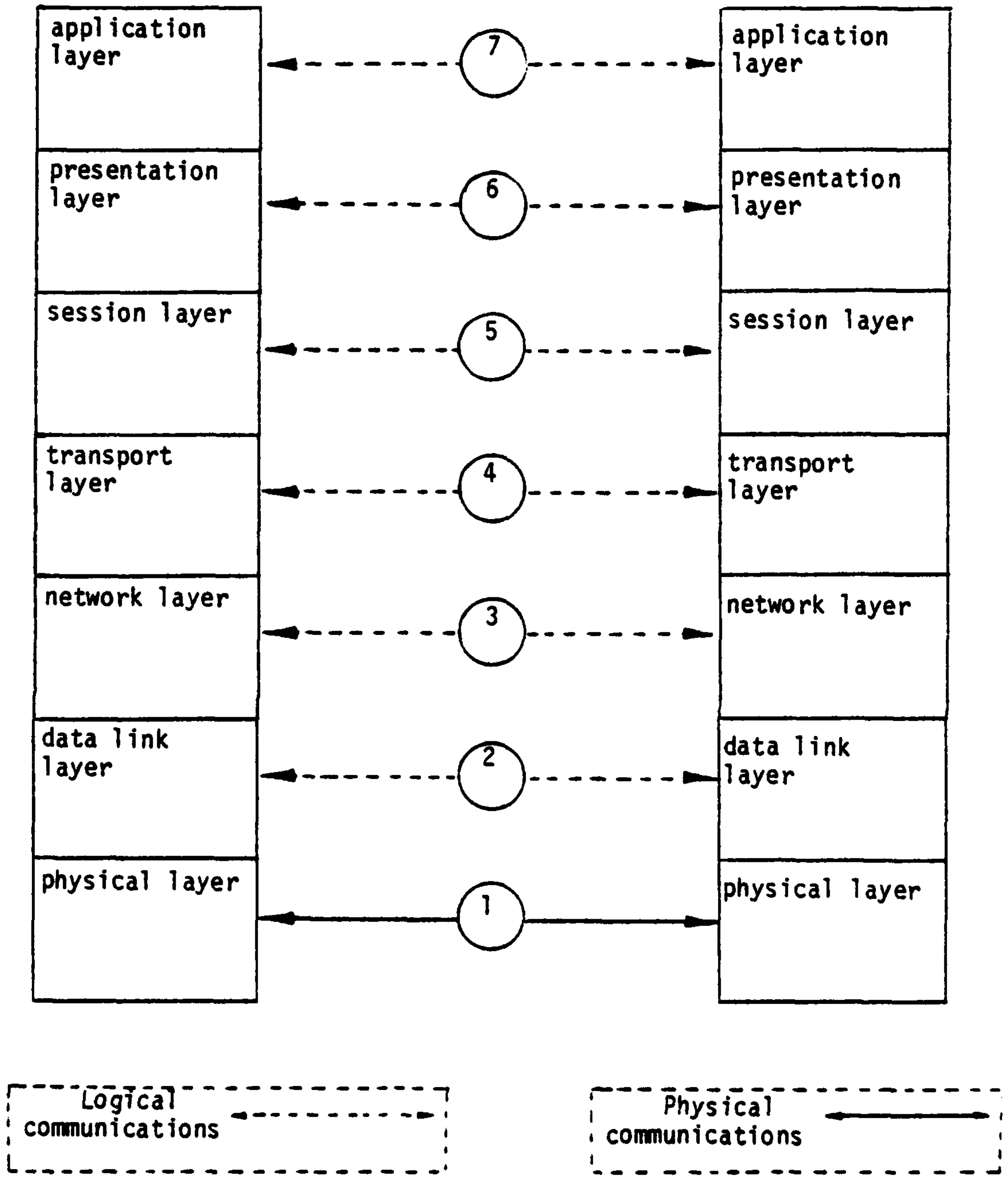


Figure 1.1 ISO/ANSI Open System Interconnection Model

Source: desJardins, R., & White, G., "ISO/ANSI Reference Model of Open Systems Interconnection," <u>IEEE 1980 Trends & Applications: Computer Network Protocols</u>, IEEE Society, pp. 47-53, Figure 3.

5. <u>Network Layer (Layer 3)</u>

The network layer is responsible for a means to transport messages through a network. It provides a logical channel between any two end-points of the network.

6. <u>Data Link Layer (Layer 2)</u>

The data link layer is to provide for the reliable exchange of data between any data equipment connected by a physical link. Any protocols required to ensure the adequacy of this exchange are appropriate to the data link layer.

7. <u>Physical Layer (Layer 1)</u>

The physical layer is made up of the physical and mechanical characteristics required to activate, maintain and deactivate the physical circuits between different devices on the network.

This thesis deals primarily with the ISO/ANSI reference model levels two, three and four.

II. TOPOLOGY

Basic design decisions which are driven by the reliability and flexibility requirements of SPLICE are the network topology and physical transmission medium to be utilized by the local network. The functions of the topology and physical medium for the SPLICE LAN are to provide for a means to interconnect the network's digital devices with a high bandwidth medium in a manner which is highly reliable and allows for reconfiguration and evolution of the SPLICE configuration.

Network topology is the arrangement of digital devices, called nodes, relative to the interconnecting media. In the recent evolution of local computer networks several topologies have emerged. This pattern of interconnection determines the distribution of network communications functions and impacts on potential reliability. And, as discussed in the following paragraphs, there is also an impact on the ease with which a network can adapt to changes in the configuration.

A. UNSTRUCTURED TOPOLOGY

In the unstructured topology, the nodes are connected in an arbitrary fashion as depicted in Figure 2.1. This is the structure most often employed for long-haul packet switching networks such as the ARPANET or the proposed Defense Data Network. The major advantage to this structure is that it allows for unconstrained network reconfiguration at minimal cost for the communications paths. Any new node is simply connected to the closest old node. This structure also provides for relatively high

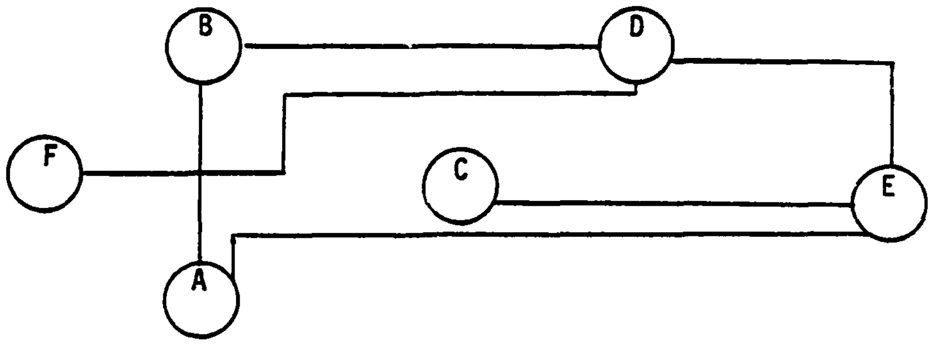
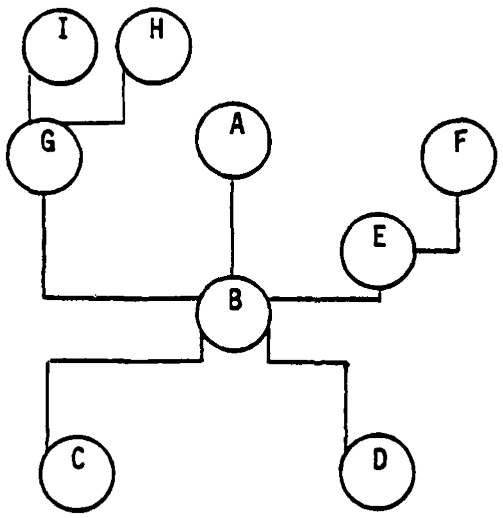


Figure 2.1 Unstructured Topology

reliability if each node is connected to two or more other nodes. The problem is that each node incorporated must contain the logic to perform switching functions or that special processors must be dedicated to the switching and routing functions. In a local network the driving issue is not the length and therefore cost of the communications path. Consequently, it is not necessary to provide costly and complex network nodes, with implementation of complex routing information, just to save on costs of communication paths.

B. STAR TOPOLOGY

The star topology is the classical interconnection technique utilized for traditional hierarchical networks. Its structure is depicted in Figure 2.2. Flexible addition of network members is a major advantage to this arrangement. Each new entity simply connects to the central switching node. Since only the central node contains switching and routing logic, all other nodes may be very simple. The drawback of this configuration is that the reliability of the local network is no better than the reliability of the central host. And, the central switch is the limiting factor for throughput achievable on the network.



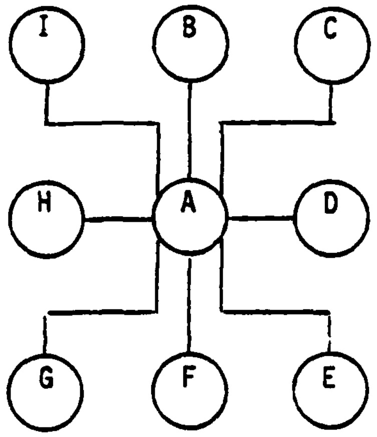


Figure 2.2 Two Variations of the Star Topology

C. RING TOPOLOGY

The ring interconnection arrangement is shown in Figure 2.3. This topology has been utilized extensively in various laboratory designs and implementations. The topology is close ended and generally only a single communications path exists between each host which are arranged in a circle or ring. Messages are passed from host to host until they complete

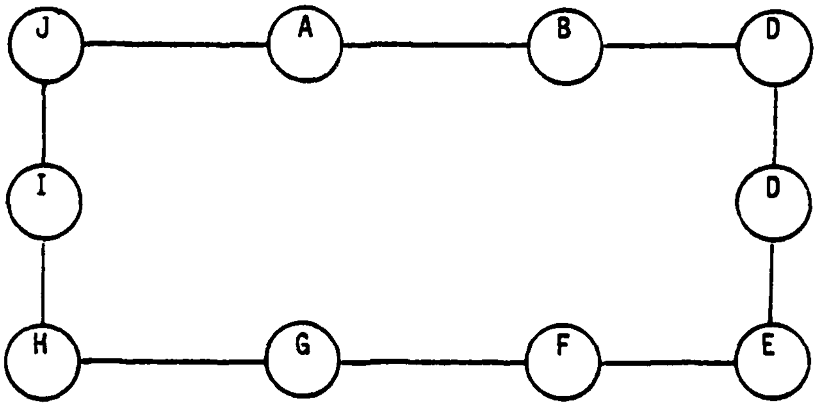


Figure 2.3 Ring Topology

the path and arrive back at their originating host. This procedure eliminates the need for routing algorithms which reduces the overhead of the network. However, each node must be operational for the network to remain fully operational. That is, when a node fails, it typically cannot pass on messages. This breaks the ring which will severely degrade the network. There are switching devices which would allow a dead host to be switched out of the network and the ring to remain intact. However, the minimum reliability of these devices then becomes the limiting factor on network reliability.

D. GLOBAL BUS TOPOLOGY

The global bus topology is indicated in Figure 2.4 This topology has also been implemented in several laboratory situations and is the structure utilized in several commercially available network packages. In this configuration, the nodes are arranged in some linear or branched configuration and connected to a common communications medium. Transmissions travel bidirectionally on the bus. Thus the system works in a broadcast mode where all nodes will receive each message simultaneously (ignoring delays due to propagation delay). Again, physical routing problems are eliminated and additionally, no single nodal failure will generally impact on the ability of other nodes to continue operation. Connection of new stations is simplified as it essentially involves only tapping on to the medium at the desired point.

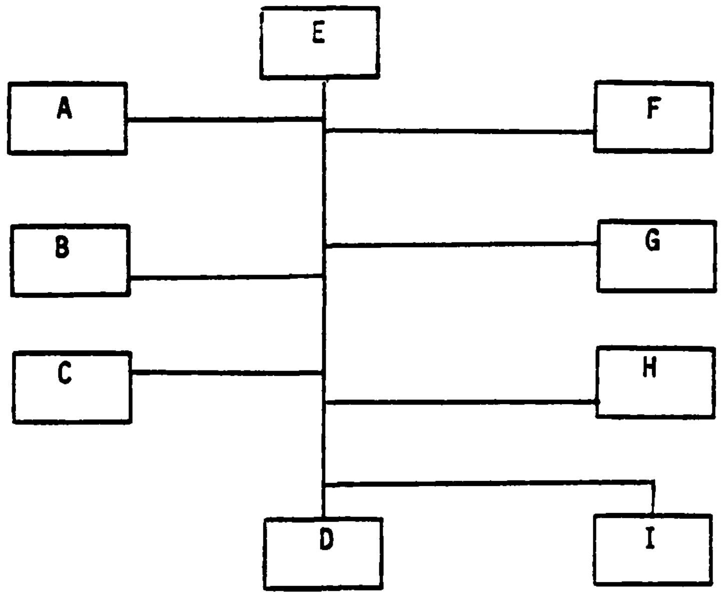


Figure 2.4 Global Bus Topology

E. CHAPTER SUMMARY

All the interconnection mechanisms described will support the communications of a local network. Thus, the key to selecting a particular topology is determined by the flexibility, reliability and simplicity afforded by a particular topology and the requirements of a potential SPLICE configuration. SPLICE [Ref. 1] reliability requirements specifically preclude a system in which the network is made inoperative by a single component failure. SPLICE configurations are also expected to be subject to change over time. Therefore, SPLICE requires a topology which provides high resiliency to single component failure while also allowing for system growth and reconfiguration. It is recommended that a global bus topology be adopted for SPLICE local networks as it satisfies both of these requirements. Protocol simplicity is also possible with this network arrangement.

III. TRANSMISSION MEDIUM

Generally, four particular technologies are considered to serve as the physical communications medium for local networks. The function of this medium is to provide a flexible, reliable and highspeed path which supports the topology selected. The four technologies considered are twisted wire pair, fibre optics, coaxial cable with base band signaling, and coaxial cable with broadband signaling.

Twisted wire pair is primarily used to support the star configuration. It is supportive of any topology in which separate communications paths between connecting devices is allowable and the required bandwidth between the devices is typically no greater than 19,200 bits per second [Ref. 7]. This technology also requires the use of digital to analog converters and modulator-demodulators to achieve maximum bandwidth. Thus, the bandwidth is fairly low and additional hardware is required to achieve the maximum.

Fibre optic technology provides a possible bandwidth of up to one gigabits per second with error rates demonstrated at one bit per billion [Ref. 8]. The transmission rate is approximately seventy per cent the speed of light and the channel is immune to electromagnetic interference. Current technology in this area does not allow for simple and inexpensive implementation of a bus topology utilizing optical cable. The problem is that fibre cable is not conducive to two-way transmission and there is no simple way of tapping onto a fibre cable. This requires bus implementation on a fibre optic medium to provide a logical bus through utilization of various reflection techniques. And, the flexibility normally afforded by a bus topology is severely impaired by inability to connect new stations to existing cable. However, an optical cable bus has been demonstrated by Kahn [Ref. 5] and Xerox is experimenting with Fibrenet [Ref. 9]. Thus, this technology may be available by the time SPLICE is fully implemented.

Coaxial cable is the primary transmission medium currently used for local computer networks. The technology has two versions: baseband and broadband.

Baseband coaxial technology places digital signals directly on the channel. It provides bandwidth to support data rate up to fifty megabits per second over distances of up to one kilometer [Ref. 10]. The error rates for this medium are typically less than one per billion, although there may be some variation depending on the exact configuration of the network. Taps which allow for device connection are inexpensive and easily installed or removed. This ease of inexpensive connectivity is a significant advantage of this technology. Whenever a node requires connection or relocation, it can be accomplished without major expense or interruption of network service. The baseband technology also provides a passive medium. That is, there is no central device which must operate to ensure that the medium can perform its function of physical data delivery.

Broadband coaxial technology, on the other hand, is based on cable television technology which is a directional broadcast system utilizing frequency division of time division multiplexing. This approach requires use of a central device which receives all messages on one frequency and then retransmits them on a separate frequency to all listening stations, or utilizes separate cables for transmission and reception. Thus, all stations transmit on one frequency and receive on another. This obviously requires the central transponder to be highly reliable and each station must connect to a frequency division multiplexor, time division multiplexor or modem in order to connect to the common coaxial cable.

The primary advantage of this technology is that it is currently available and can support twelve megabit per second data rates for distances of twenty to thirty kilometers [Ref. 11]. Additionally, the coaxial medium can be shared by other functions requiring the common medium such as closed circuit television or security alarm systems.

A. CHAPTER SUMMARY

Discussion of transmission medium technologies for utilization in the local network environment was made in the context of SPLICE requirements for reliability and flexibility. No particular technology is recommended for all SPLICE configurations. The technologies described are all capable of performing adequately in certain arrangements of hardware. The choice of a particular medium is dependent on the anticipated traffic load at a particular site as well as the distances over which the local network is expected to operate. Additionally, technology available at implementation may be a determining factor. The medium must be required to support the topology, traffic, and distances required by each SPLICE local network configuration. It should also support the reliability and flexibility inherently required in a dynamic environment. Changing the medium over time or implementation of different mediums at different sites must not be allowed to cause differences in higher level protocols which define the SPLICE local computer network.

IV. NETWORK PROTOCOLS

This chapter describes the techniques required to insure reliable delivery of messages among the processors of the local network. The underlying assumptions are implementation of a bus topology and that due to reliability considerations, the communications functions and control mechanisms of this layer should be distributed among all network nodes. This distribution would eliminate any single node responsibility for intranetwork communications. That is, a single active element failure would tend to impact only on a single node. Thus, in general, network communications can continue even with failed nodes as long as the source and destination processes are still active.

In a global bus technology all communication takes place over a shared communications channel. The channel operates in a broadcast mode where a transmission by any particular node is received by all nodes in the network. To communicate effectively via this common channel, several functions must be performed. These specific methods accomplishing the required functions must be specified and performed by all network members. The set of specifications comprises the network protocol.

The network protocol must address several problems: channel access, flow control, congestion control, priority notification, error control and recovery. The approach will be to discuss these design issues and recommend a technique to be employed in the SPLICE local area network, and, to then recommend a packet format which will support the mechanisms recommended.

A. CHANNEL ACCESS

Access control is the technique employed by a network to discipline the nodes in their attempts at accessing the communications channel. The particular access method chosen will have considerable influence on the message delay and throughput characteristics of the local network. It is the purpose of this section to discuss various methods of sharing a single communications channel and to ascertain whether a particular method is particularly suitable for the SPLICE local network configuration. Generally, channel access methods can be classified into three mutually exclusive categories: selection, reservation, and random access.

1. Selection Techniques

In all selection techniques, a particular node may access the channel only when it has somehow been signaled that its utilization of the channel is "authorized." In this system a node must save or queue all messages until its turn to use the channel. The determination of when a node "owns" the channel is not known by the node or fixed in time. Rather, a node must wait for its selection signal.

The selection can be centralized where there is a central channel controller or the selection may be decentralized in which case the logic to control the selection of a node is distributed to all stations. Because of the unacceptable dependence on a single component when utilizing a central control node, only decentralized control of the bus will be considered for the SPLICE local area network. Luczak [Ref. 12] describes three techniques of decentralized selection which we will discuss briefly.

a. Decentralized Daisy-Chaining

Decentralized "daisy-chaining" requires a separate select line to be daisy-chained through each node of the network. Additionally, all stations are connected to a common "busy" and "request" line. Any node may start the chain by activating its request line. This starts a select sequence which will move from node to node in the order in which the nodes are connected to the select line. Each node in turn propagates the select signal to the next station in sequence. Drawbacks to this selection mechanism are the additional control lines required at each node and the fact that a failure of one node will cause network failure due to blockage of the select signal.

b. Decentralized Polling

Decentralized polling is a mechanism for sharing the bus in which each station selects the next station in a predefined sequence via a short control message. A station which is selected is allowed to use the bus for a specified time interval or until it has no further traffic. At the end of its channel access time, it transmits a select message to the next node in sequence.

As the number of stations on the network increases, the message delay of this system can become significant as each physical station must be selected regardless of whether it has traffic. That is, it has no way of preregistering a demand. Also, the physical address of network nodes must be maintained and utilized in the selection method. Thus, the problem with this technique is a combination of message delay when the number of physical nodes is high and the overhead of maintaining physical addresses.

c. Decentralized Independent Requests

The method of decentralized independent requests involves a technique of time-slicing a "request period" into a number of time slots corresponding to the number of stations on the network. Each station inserts its request for transmission into its assigned time slot. Each station receives this request message and based on a universal priority table, present in each node, decides whether to transmit. The highest priority station requesting transmission is selected as all lower priority stations defer. At the end of the transmission period, another request period is begun and the sequence is repeated.

This selection method requires synchronization of all nodes for the request message time slots. It also allows high priority nodes to dominate the bus. This implies that network implementors would be forced to assign a priority to all nodes. Then, somehow and forever, management would be forced to control the assignment of processes to processors relative to the priority assignment scheme. This is particularly undesirable in a situation where no particular prioritization scheme can be identified such as is currently the case with the SPLICE local area network. Additionally, changing the priority of certain processes could require reshuffling the physical location of processes and possible processors.

2. Reservation Techniques

Reservation techniques which are sometimes referred to as collision avoidance or collision reduction techniques generally use some form of time-slot assignment for each node on the network. Slots for message transmission are reserved during a specific "reservation" period. This period is typically designated to occur during some globally known and coordinated time frame. The control of such systems may be either centralized or distributed. Tannenbaum [Ref. 13] describes several systems for accomplishing a reservation mechanism. The primary aim of these techniques is stable performance at channel utilization loads in excess of 95%. The disadvantage of these methods is that they require extremely close synchronization of all processing nodes and there is difficulty when adding additional nodes to the network. Addition of one node requires the reservation time slots to be reallocated and consequently retimed in all system nodes.

3. Random Access Techniques

The third major classification of channel access methods, and by far the most commonly implemented on global bus networks is the group of techniques which allow for "demand" allocation of the communications channel. This classification of techniques generally allows a node to access the channel at any time; there is no need for the node to wait for selection, nor does the node have to reserve time slots in which to send its information groups.

Because of allowing demand access, there may be times when two or more stations demand and acquire the channel during the same time interval. This situation will generate collisions between messages which will result in garbled or interrupted receipt of the message. Of course, the network must ensure that all garbled messages are retransmitted. The management of potential collisions and the techniques for handling the retransmission of garbled messages is another responsibility of the network layer of the local area network.

4. Random Access Control

Random access control techniques are classified as either slotted or unslotted.

Slotted techniques divide time into equal duration slots which are long enough for only one fixed-length message or packet to be transmitted. This method requires all nodes to be synchronized to a master clock. When the beginning of a time slot is at hand, one or more nodes may begin transmission. If more than one node does transmit in a given time slot, then a collision occurs and all nodes involved in the collision must reschedule transmission of the packet. The transmitting nodes assume that a collision has occurred unless they receive positive acknowledgement of the transmitted packet within a network specified time-out period.

Unslotted techniques utilize the same sort of acknowledgement system as the slotted mechanisms; however, there is no division of time into slots corresponding to message or packet length. Rather, stations may transmit at any time, not just at the beginning of a time slot, and their packet sizes may vary in length.

Both slotted and unslotted techniques can be further categorized by whether the nodes are capable of listening to the communications channel and basing decisions on whether to attempt access on the knowledge gained by sensing the carrier. These listening techniques are called carrier sense multiple access (CSMA) methods.

The first of such techniques is unslotted 1-persistent CSMA. The unslotted version of this technique has a station ready for transmission sense the channel, and if it is idle, begin transmission. Should the medium be sensed busy, the station will continue to monitor and immediately upon the channel being detected idle, the ready station will transmit its packet with a probability of one. A slotted form of this technique is also utilized in which time is divided into periods, called slots, each of which is equal to the maximum propagation delay of the communications medium. When a station senses that the channel has become idle, it must wait until the beginning of the next time slot and if the channel is still idle, it will begin its transmission with probability one. This slotted technique is only a marginal improvement in collision avoidance over the unslotted arrangement [Ref. 12]. In both of these methods, if more than one station is waiting for the idle channel, then with probability of one, there will be a collision. To avoid this problem, a technique called p-persistent CSMA evolved.

P-persistent CSMA utilizes time slots equal to the maximum propagation delay for the network. When the communications channel is sensed to be idle, the station transmits its packet with some network-adjusted probability p. Typical values for p are set a 0.1 and 0.03. With a probability (1-p), the station delays one time slot and then again senses the medium. If it is still idle, the node either transmits or defers, again utilizing its probability algorithm. Should the channel be sensed busy, the node reschedules the transmission as if a collision has occurred. This protocol was designed to lower the chances of a collision immediately after the end of a transmission without incurring idle channel time.

Another method of acquiring the channel with reduced collisions is non-persistent CSMA. This technique also can be incorporated in slotted or unslotted versions. In these protocols a node which requires the channel first senses it, and if it is found idle, the node immediately transmits or, in the slotted version, transmits in the next time slot. However, should the channel be detected busy, the node simply reschedules the packet as if a collision had occurred. This protocol is the middle road between p-persistent and 1-persistent CSMA described previously. It reduces the probability of attempts at simultaneous transmission when the channel becomes idle. However, there may be some wasted bandwidth due to no station immediately transmitting when the channel becomes idle.

This access method has been shown to be particularly effective when coupled with collision detection and transmission truncation. Figure 4.1 indicates the relative performances of various random access methods with regard to throughput and channel efficiency. The pure aloha and slotted aloha techniques shown in the figure do not utilize carrier sensing.

B. ERROR CONTROL

Error control in a bus network with random access can serve many purposes. Its primary function, of course, is to detect any error which might be generated in a packet transmission and to insure that a correct packet is delivered. Errors can be detected by either the receiver or the transmitting station. Those detected by the transmitter are discussed

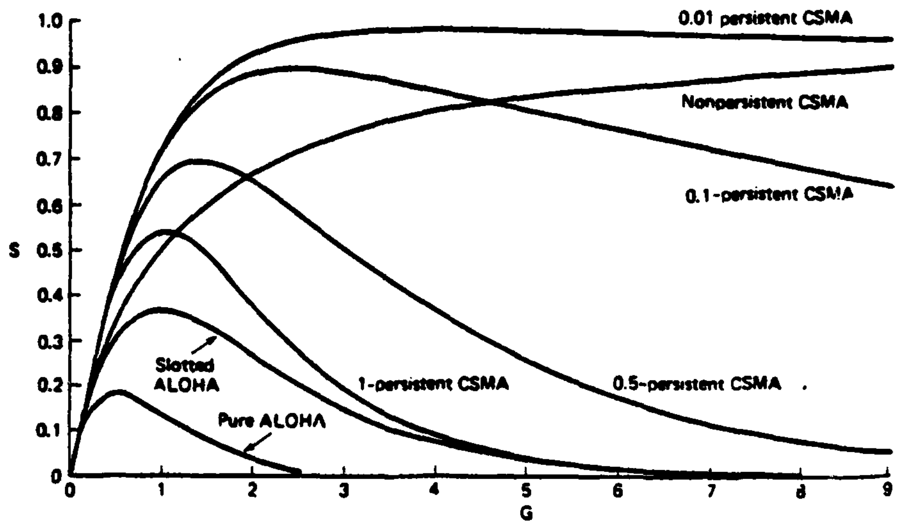


Figure 4.1 Throughput Comparison for Various Random Access Techniques

Source: Tannenbaum, A. S., "Comparison of the channel utilization versus load for various random access protocols," Computer Networks, Prentice Hall, Inc., p. 292, Figure 7.2.

as error detection while those discovered by the transmitter are discussed in terms of collision detection.

1. Error Detection

When a packet is transmitted, a cyclic redundancy checksum should be generated and its value appended to the packet. This allows a receiving node to compute the checksum and compare it to the value indicated in the checksum and compare it to the value indicated in the checksum field of the packet. If the packet is found to be in error, the receiving node discards the packet. When the checksum is correct, the receiver is required to acknowledge the receipt. This can be done with a special acknowledgement packet or can be included with a data packet bound for the appropriate node. This procedure allows for the most typical error control method utilized by contention bus networks. In this protocol, transmitting nodes assume that unacknowledged packets have been damaged or lost. That is, the transmitting node expects to receive a positive acknowledgement from the addressee of a data pcket. If such acknowledgements are not received within a specified time, an error is assumed to have occurred and the packet is rescheduled for transmission.

2. Collision Detection

With this technique a transmitting station listens to its own transmission. If a collision occurs, the reception will be garbled and the transmitting node is immediately aware of the problem. This arrangement allows for the transmission to be immediately terminated, prior to complete transmission of the complete data packet. The transmission can be rescheduled without waiting for the acknowledgement time-out period to elapse. This positive collision detection technique saves on bandwidth of the communications channel which would be lost as a result of continuing to transmit a packet which had already experienced a collision and time can be saved by allowing the transmitting station to begin collision resolution procedures prior to expiration of the time-out period.

3. Error and Collision Resolution

Once an error or collision has been detected, there must be some procedure for rescheduling the transmission. The techniques for rescheduling are classified are either adaptive or non-adaptive.

Non-adaptive methods ignore network conditions. Retransmissions are rescheduled with some algorithm which generally computes a random-delay time. This random-delay time is computed independently of the number of times the packet has been previously rescheduled. The obvious problem with this technique is that retransmissions will be scheduled in the same way regardless of the traffic conditions on the network. This does not lend itself to correction or smoothing of the network load.

Adaptive transmission techniques are tuned in accordance with current usage levels of the channel. Recent channel conditions, as derived from collisions detected, are used to dynamically adjust retransmission delays. This information can be drawn from a central control computer or the local node can handle the adjustment based on its own perceptions of the network load.

A binary-exponential-backoff technique is employed in Ethernet [Ref. 14]. With this technique, a node delays the first retransmission of a packet for a random interval with some fixed mean. With each subsequent collision of the same packet, the mean of the random delay is doubled. This has the effect of spreading out the retransmission attempts which will normally reduce the number of collisions which occur.

Geometric backoff is a similar mechanism for increasing the time between subsequent attempts at delivery of the same packet. In this mechanism for each "nth" collision of a packet, the node waits for a random period with a mean determined by "n" times "m," where "m" is the mean utilized in the previous transmission attempt. Again, this procedure has as its goal, a smoothing effect on collisions during peak traffic periods [Ref. 12].

In any case, the attempts at retransmission of a particular packet will continue until the packet is delivered or the calling process cancels the message. In the adaptive procedures, the transmissions will be scheduled less frequently with each failure to deliver. This will serve to reduce the frequency of transmission attempts to non-responsive nodes as well as to reduce transmission attempts when the network is congested.

a. Congestion Control

Congestion occurs when the number of packets attempting to utilize the network is greater than the capacity of the network. In a broadcast network, the symptom of congestion is a collision of two packets. The problem is controlled by the network through implementation of the adaptive retransmission techniques described previously. Sometimes associated with congestion control is the problem of flow control.

C. FLOW CONTROL

Flow control is a slightly different problem and involves the relationship between a transmitting and receiving node. There must be some way to keep the transmitter from overrunning the capacity of a receiver. This is a significant problem as there may be varying types and speeds of processors in a particular SPLICE configuration.

A flow control mechanism must be able to limit the number of packets which a host can send to another host, and there must be a way for the receiving host to throttle a transmitter.

The techniques for accomplishing flow control range from simple stop and wait protocols to relatively complex sliding-window protocols which allow for pipelining of packets and selective retransmission. That is, multiple packets may be transmitted to a node without waiting for an acknowledgement of each individual packet. Should some of those packets be damaged or lost, only those with discrepancies would be retransmitted.

In pure stop-and-wait protocols, a station transmits only one packet and then must wait until receiving an acknowledgement prior to transmitting another. If no acknowledgement is received, the transmitter will reschedule the packet for transmission. This technique has two drawbacks. First, it allows for duplicate packets to be passed by the network to the receiving processor. This will occur when an acknowledgement packet is damaged or lost. The second problem is that these techniques require all nodes to issue separate acknowledgements for each and every packet received. This can result in unnecessary traffic on the network and does not allow for full utilization of network resources.

A method which prevents duplicate packets and allows for multiple packet acknowledgement is the sliding-window protocol. In this type of protocol, a sending station maintains a list of consecutive sequence numbers corresponding to the number of packets it is allowed to send to a particular node. These packets are said to fall within the sending window. At the same time, receiving stations maintain a receiving window corresponding to the packet number it is allowed to receive from each node.

The packet numbers within the sender's window designate the packets which have been transmitted but not acknowledged. The set window size is the maximum number of packets which are allowed in this unacknowledged state. This number can be adjusted to allow for different characteristics of the physical pairings allowable in the network. Packets which are in the sending window may be lost or damaged in transmission and must be retained in the memory for possible retransmission. When the window is full, the transmitting node will cease transmissions to the particular node in question. The window is emptied by acknowledgements for the packets contained in the window [Ref. 15].

The receiving window tells the receiving node which packets it may accept. The window size determines the sequence of packets which are to be accepted. Packet numbers outside the window are discarded. Additionally, those below the window are acknowledged. When a packet number is inside the window, it is accepted, an acknowledgement generated, and the packet passed on for further processing.

When packets are acknowledged in sequence, the quantity of possible packet numbers to assign must be only as great as the maximum window size. If packet acknowledgements are allowed to occur out of sequence the quantity of possible packet numbers should be two times the window size, to prevent duplication of packets without danger of packet loss. Planning for potential packet number assignments is essential as it is not practical to number packets from one to infinity. Rather, it is necessary to use module n numbering where n is the upper limit of allowable packet numbers. In any case, the receiving node inspects the packet number to see if it is within the window. When a packet within the window is received, the packet number is removed from the window.

If the receiving window is maintained at one, the packets are only accepted in sequence. This can also be insured for window sizes greater than one by requiring that the acknowledgement for a packet n be generated only if packets 0 through n have been properly received. If one of the packets in a window sequence is damaged, then all packets of that window might require retransmission. This procedure is acceptable on a local broadcast network as window sizes will generally be small. Also, packets arriving out of sequence should be virtually eliminated as the transmitting station is detecting its own collisions and should never allow out-of-sequence transmissions to a particular node.

It should be noted that the packet numbers should only be assigned to messages containing data. This insures that pure control messages will always be accepted by a node. Thus, there must be some mechanism for informihg the receiver to ignore the window. This is done by indicating a specified value in a "packet type" field of the packet header. The value of "packet type" will indicate whether the packet contains data and the "packet number" field is significant or should be ignored.

1. Implications of a Flow Control Window

In order to utilize a window mechanism for flow control, each node must maintain certain information concerning packet and window status for each node. This information could be realized in the form of a "communications control block" as represented in Figure 4.2. This table could also contain other information such as pointers to packet queues or transmission sequence information.

This particular mechanism will allow for a node A to transmit a packet with sequence number "one" to a node B. Then node A could transmit a different packet also with sequence number "one" to some node C. The packet numbers only have meaning within a node-pair relationship. This allows for the window sizes to vary from one node pair relationship to another.

	Node a	Node b	Node c
next to send			
next to receive			
retransmission attempt			
timer start			
send window			
receive window			
initialization information			
other information			

Figure 4.2 Possible Communications Control Block Information

Requiring each node to maintain a communication control block with information such as "next to send" and "next to receive" implies that there must be some mechanism for starting over if a node should somehow lose its status information.

It is proposed that a field be provided in the packet header to indicate a reset of status information. This would have the effect of re-initializing status information for a particular node throughout the network. For example, Node A loses its status information (Node A crashes). When this is discovered by Node A (generally when it once again becomes operational) it generates a reset message to all nodes on the network. Each node would re-initialize the values in its communications control block which were particular to Node A. Node A would, of course, re-initialize all of its communication control block values. The reset message should be transmitted a set number of times prior to Node A resuming network operation. This will insure that every operative node has a chance to properly receive the reset. Only nodes which were not operative would miss the reset. These nodes would be required to generate their own reset as they resume network operations.

D. INTRANETWORK MESSAGE FLOW

The network management will transport packages from node to node. However, there are additional requirements for control which must be incorporated into the network protocol. These features are to allow for priority message notification and end of message indication to insure that a receiving host properly responds to the sending process. These mechanisms are often accomplished by generation of separate packet types with distinct headers. But, it is proposed that one general header can accomplish all required functions. This will generate some additional packet length, but at the same time will allow piggybacking of control information in a data packet whenever required. The following description of the packet format as depicted in Figure 4.3 indicates a technique for accomplishing node-to-node communication within the local network. The features suggested are not intended to be the only functions which can be performed by the network. Other features could be incorporated in a similar manner as they are required. The following discussion is a description of the packet format fields of Figure 4.3.

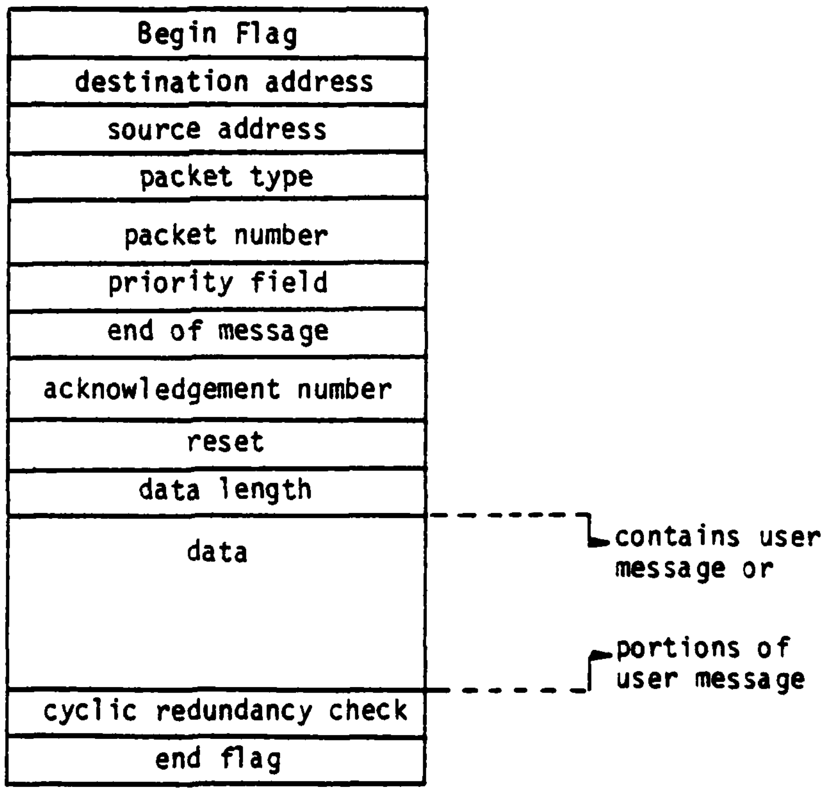


Figure 4.3 Packet Format

E. PACKET FORMAT

The function of each field of figure packet is discussed in the following paragraphs. No specific length for the fields is recommended as this should be decided only after specific network configurations and hardware specifications have been determined. It is recommended that all control fields be of a fixed length with only the "data field" allowed to vary.

1. Flag Field

The beginning flag field is required to indicate to all processors on the network that a message is being transmitted. In addition to marking the beginning of a message, the flag field is used to synchronize the receiving processor with the incoming bit stream. While no specific flag is recommended, it should be chosen such that its length is sufficient for positive identification and its distortion due to collision with another transmitter's flag is easily discerned by the collision detection hardware. The use of the flag sequence in data or control information between flags must be prevented by use of a bit stuffing mechanism. The end flag marks the end of the packet.

2. Destination Address

The destination address field serves the purpose of informing the correct node to copy the rest of the bit stream and continue processing. A unique physical address for each node is recommended in order that address recognition can be inexpensively implemented in hardware.

3. Source Address

The source address is required for proper addressing of acknowledgements and communications control information which must be maintained by each node. The network layer can very simply construct its acknowledgement by reversing the source and destination addresses and appropriately setting the acknowledgement field. It also uses the source information for updating its communications control block.

4. Packet Type

The type field indicates whether the packet contains data or consists of only header and trailer fields. This feature allows the network to pass control packets independent of the window control feature. Control packets may indicate acknowledgement or reset situations.

5. Packet Number

The packet number is the sequential number of the packet transmitted to a particular node. The receiving node utilizes this number to determine whether the packet is a duplicate and whether the packet should otherwise be accepted for processing. This packet number must be drawn from a generation technique which provides enough numbers to adequately cover the maximum window size of the network. For a window size of one, this packet number could be a one bit field with values zero or one. That is, Node A transmits packet 0 and increments "next to send" to 1. Node A transmits sequence 1 and increments next to send to 0. If the received packet number is larger than expected, the packet is ignored. When the packet number received is less than the number expected, the receiver generates a acknowledgement for the packet and then discards it. The packet number would only be below the expected value when the sending node had failed to receive an acknowledgement for a packet which had already been accepted by the receiver. This procedure will assure sequential delivery without duplication.

6. Priority Field

The priority field is provided in order for using processes to indicate that a message to a particular node should be brought to the receiving node's attention, prior to normal message handling. The value for this field must be passed as a parameter when the network is called upon to send a message.

7. End of Message Field

The end of message field is used to signify the last packet of a process's message. This end of message is used to signal the receiving node that a particular packet is the end of a logical sequence.

8. Acknowledgement Field

The acknowledgement field is required for acknowledging receipt of a packet. The value of the field is the number of the last packet correctly received. The acknowledgement portion of the local network header should contain the packet number of any packet which is being acknowledged. If a packet contains no acknowledgement, the value of this field should be set to some impossible value which will be ignored by the receiving station.

9. Reset Field

The reset field is provided to indicate that a sending node requires reinitialization of all network level communications control information as discussed in the earlier recovery section.

10. The Data Length Field

The length of the data field is variable as thus must be indicated for each packet sent. This knowledge allows a receiver to determine the proper location of remaining control fields.

11. The Data Field

The data field contains the information to be delivered by the network. All other portions of the packet header are stripped from the packet when it arrives at the receiving node.

12. The CRC Field

This field contains the cyclic redundancy checksum value as computed for the packet by the transmitting node. The receiving node compares the value found in this field to the checksum it computes for the received packet. A match indicates no errors so the packet is accepted and acknowledged. If an error is indicated, the packet is discarded and not acknowledged.

F. CHAPTER SUMMARY

Various protocols for managing a bus topology were examined. Because of reliability, simplicity, and suitability to interactive or bursty communication, a random access contention mechanism with collision detection is recommended for the channel access method to be used in the SPLICE local area network. Based on this, procedures for controlling errors, network congestion, traffic flow, and recovery were discussed. Finally a packet format was presented which will support the recommended protocols. A hierarchical input-processing-output (HIPO) overview of the modules required to implement this network protocol appears in Appendix A.

V. END-TO-END PROTOCOLS

The network management techniques recommended provide a virtual circuit for the communications between processes. The network will deliver data in the sequence in which it is introduced to the network. However, due to special considerations which may arise, there is need for a higher level protocol between the using processes and the network. These special considerations are brought about by the fact that a user process's message length can generally be greater than the length which can be optimally handled by the network. Long-haul networks typically define a maximum message length which is subsequently subdivided by the long-haul network's protocols. Thus, if a user's message is longer than this maximum length, there must exist some technique for subdividing the message into a length which will be acceptable to the long-haul interface. For messages which are remaining within the SPLICE local network, there also may be some upper limit on the size which is allowed for a single transmission. The maximum size may be determined by the number of bits which can be transmitted in some maximum allowable transmission length. Or, more likely, the maximum size will be determined by the number of bits which can be handled by buffers dedicated to message reception. In either case, there must be some mechanism to perform the required division of messages which are too long, and to track the pieces to ensure that all portions of the message arrive at the destination process. This mechanism is normally identified as an end-to-end protocol.

Since it is not our intent or purpose to re-invent the wheel, we will now look at one end-to-end protocol that has been developed which could be implemented in the SPLICE local area network to satisfy the requirements mentioned above, and ensure end-to-end delivery of messages regardless of the network or number of networks which are traversed by a particular message. This end-to-end protocol is the Advance Research Project Agency's (ARPA) Transmission Control Protocol (TCP).

A. ARPA TCP

TCP's goal is to provide a fully reliable virtual circuit service across many networks. Full error detection and recovery procedures are included in TCP with the anticipation that some networks in a multi-network system might not provide full reliability [Ref. 16].

To allow fine-grained flow control and more efficient error recovery, TCP sequence numbers are in units of octets. This also allows repeated fragmentation of data into arbitrary-length packets, if necessary, between networks with varying packet sizes. The user interface to TCP is record oriented. That is, complete messages are passed to and from interfacing processes but error and flow control are performed on an octet basis.

While the original TCP had long sequence numbers and highly reliable procedures for data transfers once a connection was established, the procedures for first establishing a connection were rather simple. Several weaknesses in these connection management procedures were identified in the course of TCP development, which lead to the invention and adoption of more sophisticated techniques [Ref. 17]. In particular, techniques for initial sequence number selection, the three-way handshake for connection establishment, and resynchronization for clock based sequenced numbering were proposed and implemented. These enhancements have made TCP one of the most sophisticated and complex end-to-end protocols ever developed [Ref. 16]. And, it is designed to deal with unreliable subnets and the delayed duplicate problem via the three-way handshake as shown in Figure 5.1 which works as described in the following sections.

B. TCP THREE-WAY HANDSHAKE

1. Normal Operation

Normal operation is depicted in Section A of Figure 5.1. In this mode, A chooses a sequence number X, and sends it to B, as indicated by the two rightward arrows. B replies with a SYN 2 message acknowledging X and announces its own initial sequence number, Y (which may be equal to X). Finally, A acknowledges B's choice of an initial sequence number in its first data message.

2. Delayed Duplicates

The following discussion refers to Part B of Figure 5.1. The first message is a delayed duplicate SYN, from a connection since closed. This message arrives at B without A's knowledge (shown by ... in A's column). B reacts to this message by sending A a SYN 2 message asking for verification that A was indeed trying to set up a new connection. When A rejects B's attempt to establish, B realizes that this message was a delayed duplicate and abandons the connection.

#	A	Packet	B	Comment
1	→	[Type=SYN1, SEQ=X]	→	A wishes to initiate
2	←	[Type=SYN2, SEQ=Y, ACK=X]	←	B accepts A's request
3	→	[Type=DATA, SEQ=X, ACK=Y]	→	A acks B and start transmission

(a)

#	A	Packet	B	Comment
1	...	[Type=SYN1, SEQ=X]	→	Delayed duplicate SYN1 arrives at B
2	←	[Type=SYN2, SEQ=Y, ACK=X]	←	B accepts A's request
3	→	[Type=REJECT, ACK=Y]	→	A rejects B's connection

(b)

#	A	Packet	B	Comment
1	...	[Type=SYN1, SEQ=X]	→	Delayed duplicate SYN1 arrives at B
2	←	[Type=SYN2, SEQ=Y, ACK=X]	←	B accepts it
3	...	[Type=DATA, SEQ=X, ACK=Z]	→	Delayed duplicate B sees that $Z \neq Y$
4	→	[Type=REJECT, ACK=Y]	→	A rejects B's attempt as before

(c)

(a) Normal Operation

(b) Delayed SYN1

(c) Delayed SYN1 and delayed acknowlegement

Figure 5.1 TCP Three-Way Handshake

Source: Tannenbaum, A. S., Computer Networks, Prentice Hall Inc., p. 350, Figure 8-10.

3. Delayed Duplicates, Worst Case

This case, as shown in part c of Figure 5.1, occurs when both a SYN1 and an acknowledgement to a SYN2 are floating around in the subnet. As in the previous example, B gets a delayed SYN1, and replies to it. At this point it must be realized that B has proposed using Y as the initial sequence number for B to A traffic, knowing full well that no messages containing sequence number Y, or acknowledgements to Y are still in existence. When the second delayed message arrives at B, the fact that Z has been acknowledged rather than Y tells B that this, too, is an old duplicate.

C. DEFENSE DATA NETWORK CONSIDERATIONS

Although AUTODIN II has been replaced by the Defense Data Network (DDN), the basic transport protocol utilized by AUTODIN II will also be utilized by the DDN. The protocol is TCP. While this in itself does not require that the SPLICE local network be interconnected through the DDN, we feel this possibility should be considered in any recommendation for an end-to-end protocol.

D. CHAPTER SUMMARY

It should be apparent from the above discussion that TCP has all the capabilities required for internetwork communication and provides good error detection and recovery mechanisms. Also, even though AUTODIN II is dead, the Defense Data Network is planned, and in our opinion will eventually be used to interconnect the SPLICE local network configuration. Thus, any computer communication network interconnected with the proposed

DDN must either implement the TCP for internetwork communication, or correct for any disparities between its own host-to-host protocol and the TCP at the interface to the DDN. If TCP is used as the primary host-to-host protocol in a network interconnected with the DDN, internetwork communication will be simplified. However, due to its complexity, TCP may not be well suited for intra-local network communications. The remainder of this thesis looks at this possibility in depth.

VI. SUITABILITY OF TCP

TCP facilitates the transmission and reception of messages between host computers which reside in an internetwork environment. The internetwork environment is assumed to consist of several host computers connected to a local network which is in turn connected via gateways to other computer networks [Ref. 18].

TCP is an end-to-end reliable protocol designed to fit into a layered hierarchy of protocols which support multinetwork applications. It interfaces on one side to a lower level protocol such as the Internet Protocol. (The Internet Protocol provides facilities for transmitting blocks of data, called datagrams, from source to destination in different networks and fragments packets when the adjoining network will support only a smaller packet. Thus, the internet protocol could have little or no effect on intranetwork communication performance) [Ref. 18].

Other aspects of TCP are features designed to efficiently utilize the communications facilities in long haul networks in which messages are routed to their destination via a multi node path. For this type of network, TCP has been designed to give low message delay, high throughput, and provide for cost-effective use of the network resources, particularly channel capacity. The SPLICE local area network, however, is a local network with features very different from those mentioned above. Delay, throughput, and cost considerations may be better satisfied by using different techniques than are contained in the TCP as it is presently specified.

A. TCP FUNCTIONS

TCP accepts messages from host processes and performs the following basic functions:

- Breaking down of message into packets.
- Sequencing and reassembly of received packets.
- Retransmission of lost or incorrectly received packets.
- Flow control.
- Detection of duplicate packets.

1. Message to Packet Conversion

TCP accepts a message from a host process and depending on the length, divides it into one or more packets. These packets are relayed in a store and forward fashion until the destination node is reached. This provides for improved message delay and lower network costs in a long haul network. Packet switching decreases message delay by providing for pipelining of packets [Ref. 19], and it utilizes network resources in a cost-effective manner by efficiently utilizing telephone line capacity. With packet switching, each communication channel is shared by many connections, rather than dedicating a single channel to a single connection. Message switching shares communication channels also, but it does not offer improved message delay achieved by pipelining of packets over a multi-hop path.

In addition to providing low message delay, and efficient use of bandwidth, packet switching in a long haul network allows packets to arrive out of sequence, thus requiring resequencing to be performed by the destination TCP.

From the standpoint of message delay and throughput, packet switching is not well suited to the SPLICE local area network's intranetwork communications, since the SPLICE local area network is to be a collection of nodes connected by a high speed bus, whenever a node on the local network transmits to another node also on the local network, a single-hop connection is established and there is no requirement for employment of a store and forward technique. Since no opportunity for multiple path transmission exists, packet switching technology is not appropriate for a local area network bus structure. Analysis of Ethernet, a local area network using contention bus, reveals that the network efficiency (the fraction of time the bus is carrying good packets) increases as packet length increases [Ref. 20]. Thus, low message delay, efficient channel utilization and protocol simplicity are all achieved if very large packets are used. This would normally involve sending the message to the destination intact which in turn would eliminate the complexity and degradation in performance that would result from unnecessarily breaking messages into smaller packets and reassembling them at their destination. This would also decrease overhead in the local network by requiring only a single header for the entire message rather than a single header for each packet of a message. An exception to the above argument would only occur when a user process had an excessively long message that exceeded the destination's communications buffer size or the length was such that the transmission time required to send the message would exceed some upper limit imposed in order to manage bus contention.

2. Sequencing and Reassembly

Since packet switching allows packets to arrive out of sequence, each octet of data in a packet is assigned a sequence number at the source TCP. These sequence numbers are used to resequence the packets at their destination TCP prior to being reassembled in a message. The need for resequencing in the SPLICE local network is eliminated since alternate routing cannot occur over a bus. The protocols recommended in the previous chapters of this thesis insure sequential delivery within the local network.

3. Flow Control

The DOD TCP protocol uses a window technique for flow control. The receiving TCP sends to the source TCP a window of sequence numbers which it will accept with every acknowledgement. As messages are received, or octets in the case of a stream mode, they are checked to see if they fall within the acceptable range of sequence numbers. If they do not, they are discarded. As messages are received and discarded, the window is shifted at both the destination and source TCP's so that new sequence numbers can be brought into the acceptable range.

This use of the window technique enables a destination TCP to regulate incoming packet flow, thereby preventing the receipt of more packets than it can accommodate. During periods of non-congestion, the use of a large window allows packets to be sent from the source well before the destination TCP is ready for them, resulting in an uninterrupted flow of packets. This yields higher throughput than would result from a small window [Ref. 20]. Since round trip delay over a long haul network is typically an order of magnitude greater than the delay caused by communication protocol processing, a simple protocol that waits for acknowledgement of each packet before sending the next will cause communication channels to be idle a large percentage of the time, resulting in poor throughput. However, in a local network, communication channel capacity is not as significant a limitation to throughput. Adequate throughput can be achieved if flow control is implemented such that a message is not transmitted from a source end-to-end protocol until the destination end-to-end protocol acknowledges the previous packet. Thus, with this latter technique, protocol simplicity, and an increase in throughput resulting from decreased protocol processing delay, can be achieved by eliminating all of the various computations required to check incoming packets to see if they fall in the window, to shift the window, and to recompute the window size as network conditions change. Simplicity is also achieved since the need is eliminated for mechanisms which ensure that both the sending and receiving TCP's windows are synchronized.

We feel that the flow control recommended for the network in Chapter 4 is adequate for intra-local network communication. The TCP flow control is required only for traffic which must pass between nodes which are on separate SPLICE configurations.

4. Detection of Duplicate Packets

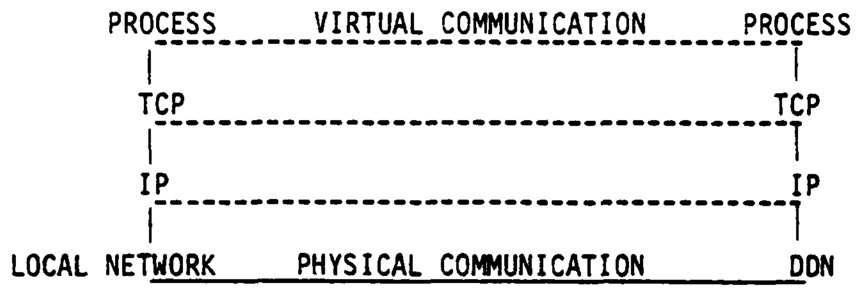
The TCP also uses the window technique for detection of duplicate packets. When a duplicate arrives, it will of course be outside the window of acceptable packets and be discarded. However, with the simpler flow control technique suggested for the local network communications, a receiving TCP will be expecting only a single packet sequence number, the one immediately following the number of the preceding packet. Thus, a receiving TCP can detect duplicate packets by comparing its sequence number to the one it expects to receive. Again, TCP provides more power than is required for intra-local network communication while satisfying the requirements for reliable communication for inter-SPLICE data traffic.

B. LOCAL NETWORK TO SPLICE NATIONAL NETWORK CONTROL

We have attempted to show by the preceding discussion that the TCP protocol, as it is currently specified, is more complex than necessary for use in a local area network such as the SPLICE local area network. Furthermore, measurements of the TCP [Ref. 20] indicate that it has very poor throughput compared to a high speed (10 Mbits/sec) bus such as will be used in the SPLICE local network, and therefore will not provide optimal local network communication performance. However, since the SPLICE local network is to be interconnected with the Defense Data Network sometime in the future it must be able to communicate with TCP. This leaves us with several design alternatives which are described in the following section.

C. DESIGN ALTERNATIVES

First, the local network could implement TCP for communications in both the local network and for communications via the DDN as depicted in Figure 6.1. This approach would provide for relative ease of communication with hosts residing on the DDN and eliminate the need to



------ VIRTUAL COMMUNICATION

——— PHYSICAL COMMUNICATION

IP is for other (non-DDN) nets

Figure 6.1 Alternative One

develop a node-to-node protocol specifically designed to provide good performance in the local network. However, the simplicity of this strategy is achieved at the expense of suboptimal performance. This problem could be overcome, however, by modifying TCP in such a way as to simplify the processing of messages which originate at and are destined for processes which reside within the local network, yet still provide the capability required for communication between a process which is in the local network and one which resides outside. This dual capability requirement could be accomplished by implementing the local network transport protocol so that certain functions of TCP could be either executed or bypassed, depending on the situation. This assumes that each host can identify which hosts reside within the local network and those which reside outside.

The major advantage of this technique is its simplicity. This is important if a large amount of traffic will be traveling through the network. Additionally, an increase in processing time will generally result in the requirement for increased buffer space to hold packets which await processing.

The disadvantage of this approach is that it restricts the amount of tailoring which can be designed into the local network transport layer for communications between hosts which reside within the SPLICE local area network. Furthermore, the message formats and various protocol conventions which are used in the TCP must be adhered to.

A second approach is shown in Figure 6.2. This approach would enable the local network transport layer to posses only those functions required for communication with other transport layers within the local network. Those extra functions required for communication with TCP's outside of the local network would be handled at the front-end processor (location of the translator).

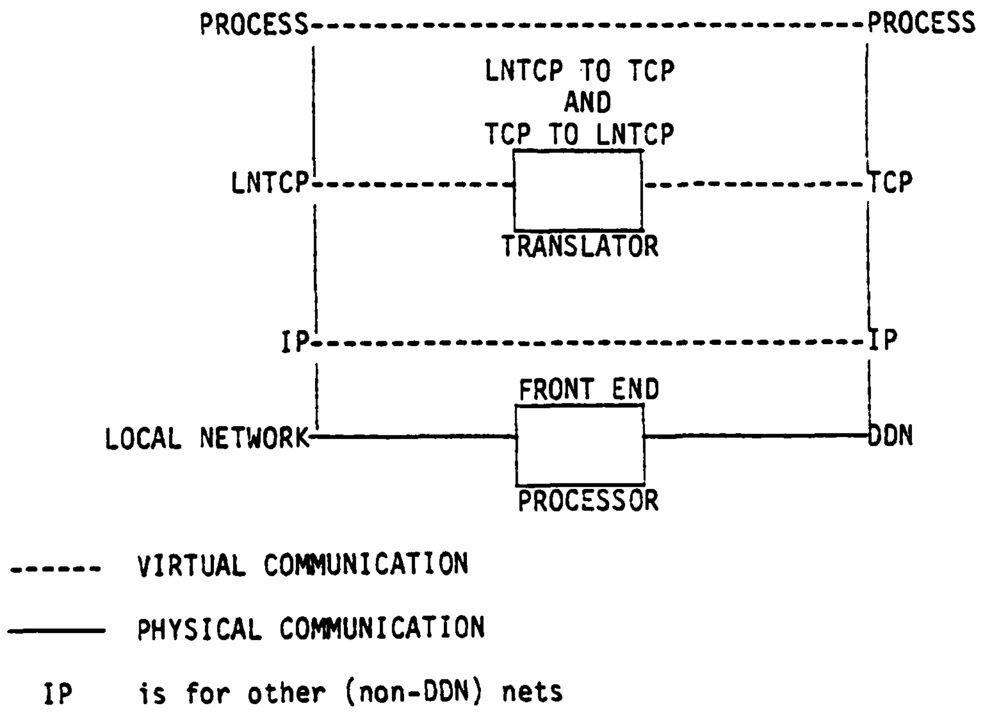
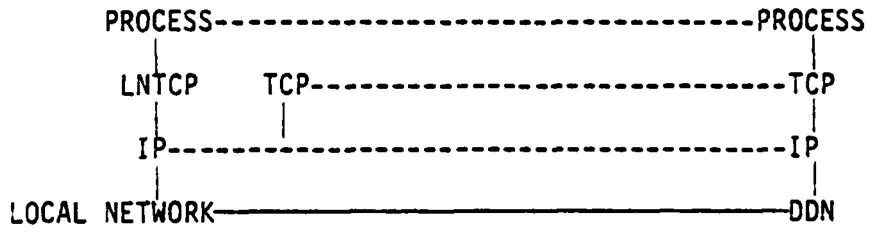

PROCESS
PROCESS
LNTCP TO TCP AND TCP TO LNTCP
LNTCP
TCP
TRANSLATOR
IP
IP
FRONT END
LOCAL NETWORK
DDN
PROCESSOR
------ VIRTUAL COMMUNICATION
——— PHYSICAL COMMUNICATION
IP is for other (non-DDN) nets

Figure 6.2 Alternative Two

The code for DDN communication could be eliminated at each host. The functions eliminated would be implemented at the front-end processor in the form of a translator which would convert local network messages to a format suitable for DDN communications. This approach has the disadvantage of requiring more processing at the front-end processor. The advantage is, implementation is simplified at each local network host. This probably would result in increased performance of hosts within the local network, over the previous strategy, and would result in a cost savings particularly for a large number of hosts.

A third approach to the implementation of the local transport layer is shown in Figure 6.3. With this strategy, each host site will contain both the TCP and the local network end-to-end mechanism. The local transport layer would be reserved for intranetwork communication and TCP would be reserved for internetwork communication.



------ VIRTUAL COMMUNICATION

——— PHYSICAL COMMUNICATION

IP is for other (non-DDN) nets

Figure 6.3 Alternative Three

This technique, as with the first approach, is relatively simple which also allows for the use of a local transport layer which is designed specifically for intranetwork communication. The disadvantage is the cost required to maintain two end-to-end protocols at each network node.

A final solution would be to implement TCP in the same manner as the Worldwide Military Command and Control System (WWMCS) in its WWMCS Intercomputer Network (WIN) configuration. The primary advantages of this technique is that it has been designed to interface with ARPANET (DDN will resemble ARPANET). The primary disadvantage of this approach is that performance would not be as good as the previously discussed alternatives. Although this was not discussed, we feel this is an interesting option which might merit additional research and have included a rough draft functional description of the system proposed by Digital Technology Incorporated as Appendix B [Ref. 21].

D. CHAPTER SUMMARY

TCP is a complex protocol with features designed to efficiently utilize the communications facilitated in a long-haul network such as the proposed DDN. Any local network that uses TCP as its only end-to-end protocol will achieve relatively simple internetwork communication at the expense of suboptimal intranetwork performance. An alternate approach would be to implement a local network transport protocol which retains many of the features of TCP, but substitutes for the TCP flow control, resequencing, and duplicate detection features, and which uses a much larger packet than used in packet switching networks. Such a protocol should provide for efficient "intra" and internetwork communication performance. Several design strategies have also been presented with the basic differences between them being complexity, storage requirements and processing time. Our recommendation is the approach depicted in Figure 6.2. In this approach, the local transport control protocol is so specified that only those parts of TCP which are necessary for the local area network are included.

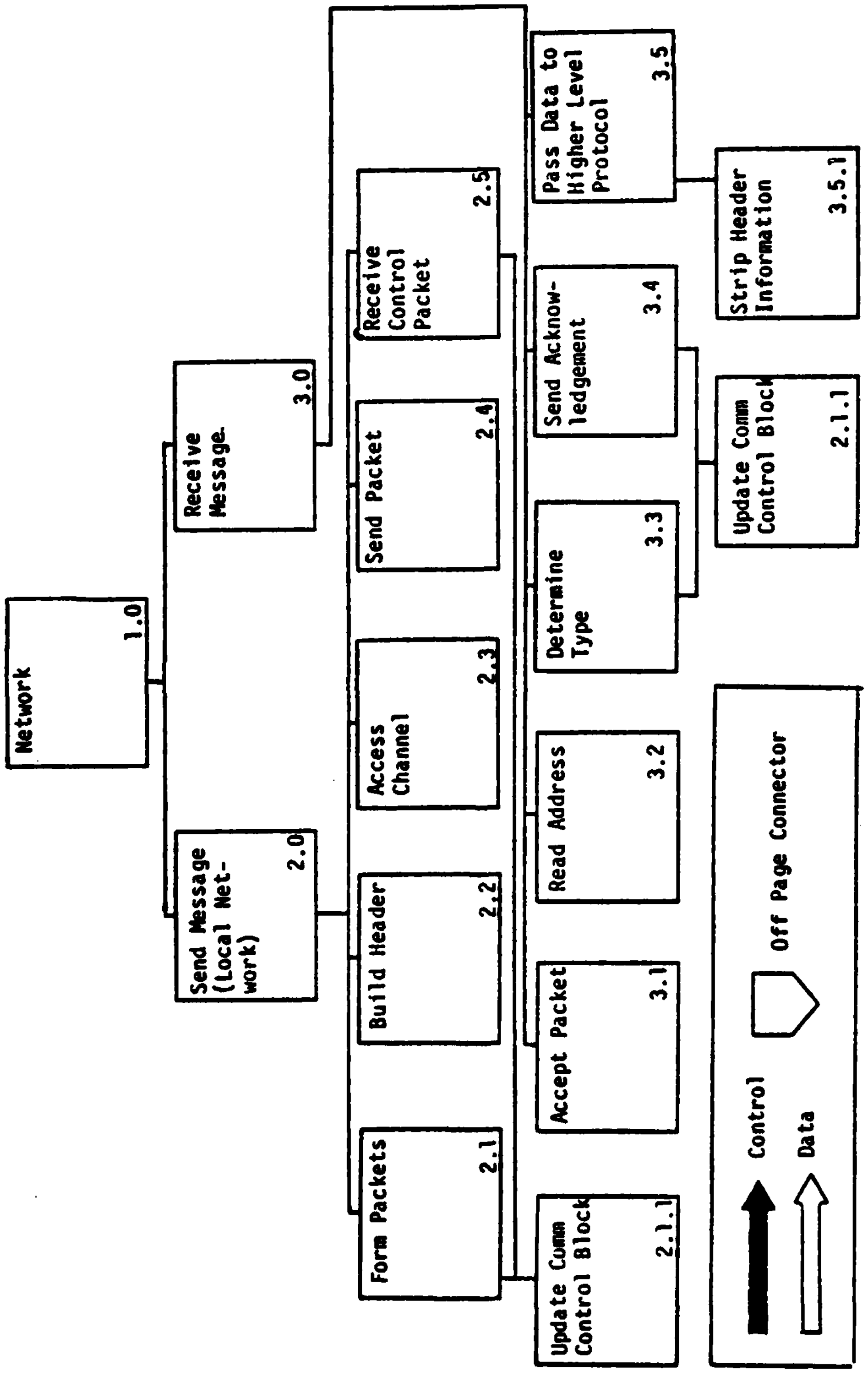
Since the SPLICE request for proposal (RFP) [Ref. 22] requires compatibility with X.25, a description of this protocol appears in Appendix C for the reader's information.

VII. RECOMMENDATIONS

This thesis has examined design considerations for the transport and network layers of a local area computer network for SPLICE configurations. These examinations have been conducted with the basic design decision being that reliability considerations are best served through implementation of a bus topology. The thesis recommendations are summarized as follows:

- Utilize a bus topology for the SPLICE local area computer network.
- Utilize a transmission medium that will support the bandwidth, expansion, and distance requirements of each SPLICE local network configuration.
- Implement a random access contention mechanism with collision detection to allocate the communications medium.
- Implement network management techniques and packet formats to support the recommended access method. Chapter 4 provides examples and approaches to this management problem.
- TCP should be utilized for both the SPLICE local area network intranetwork and internetwork communications. However, for intranetwork communications, TCP should be so specified that only those parts necessary for the local area network are included.

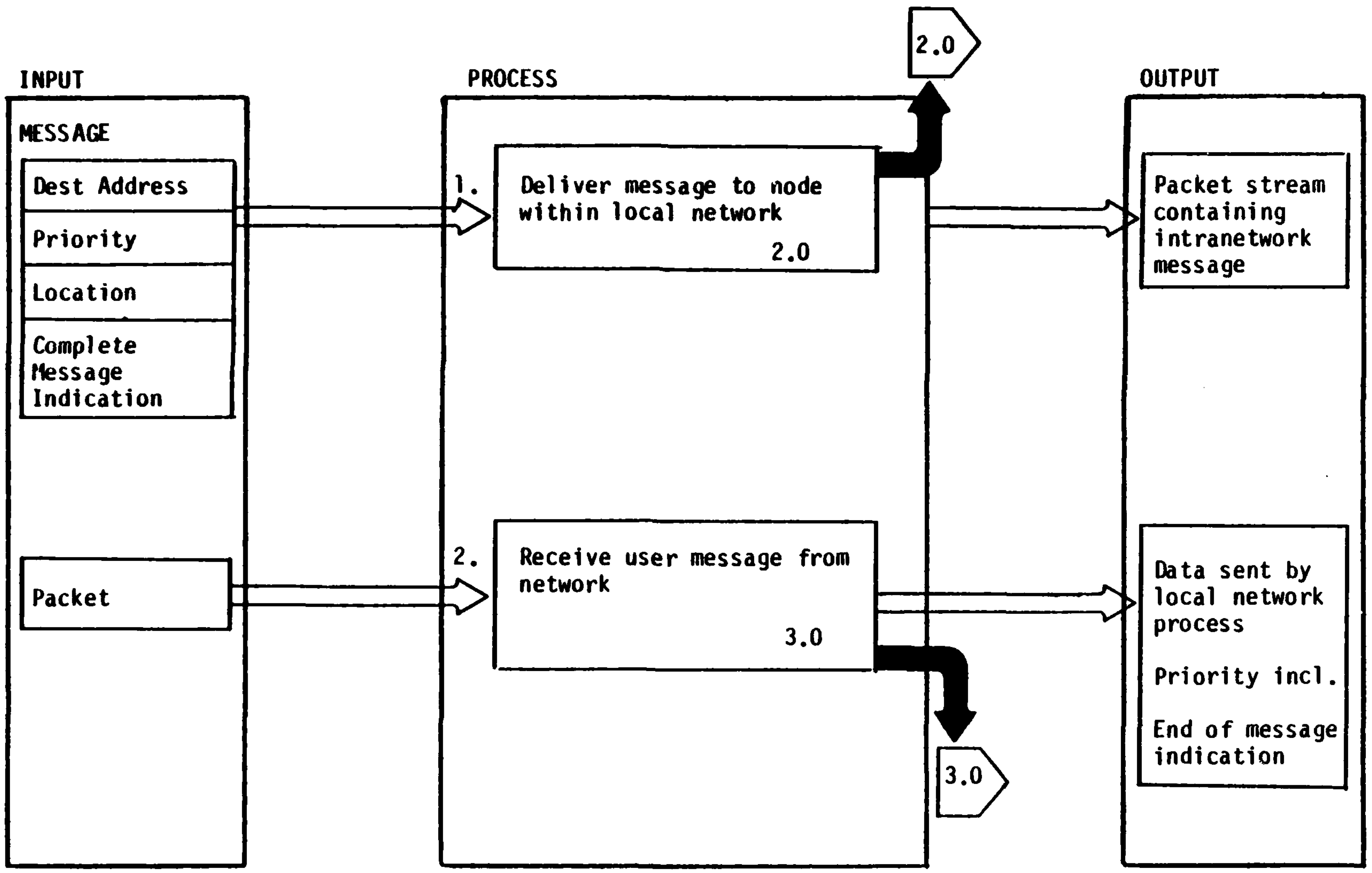
APPENDIX A: Hierarchical Input Processing Output for the Network Layer



Visual Table of Contents

SYSTEM/PROGRAM Network Protocol DIAGRAM ID 1.0

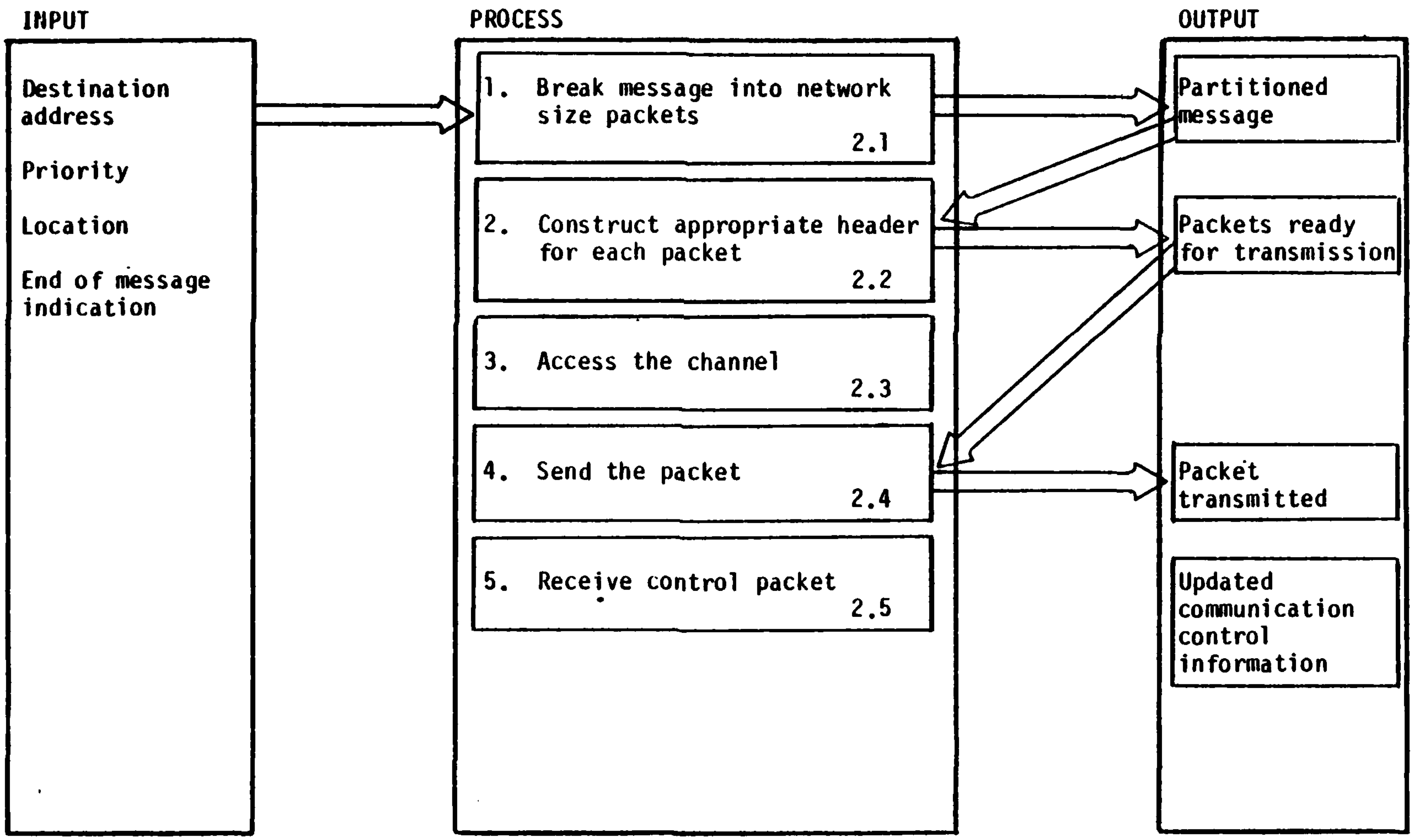
NAME Intranetwork Communications



INPUT
MESSAGE
Dest Address
Priority
Location
Complete Message Indication
Packet
PROCESS
1.
Deliver message to node within local network 2.0
2.
Receive user message from network 3.0
2.0
3.0
OUTPUT
Packet stream containing intranetwork message
Data sent by local network process
Priority incl.
End of message indication

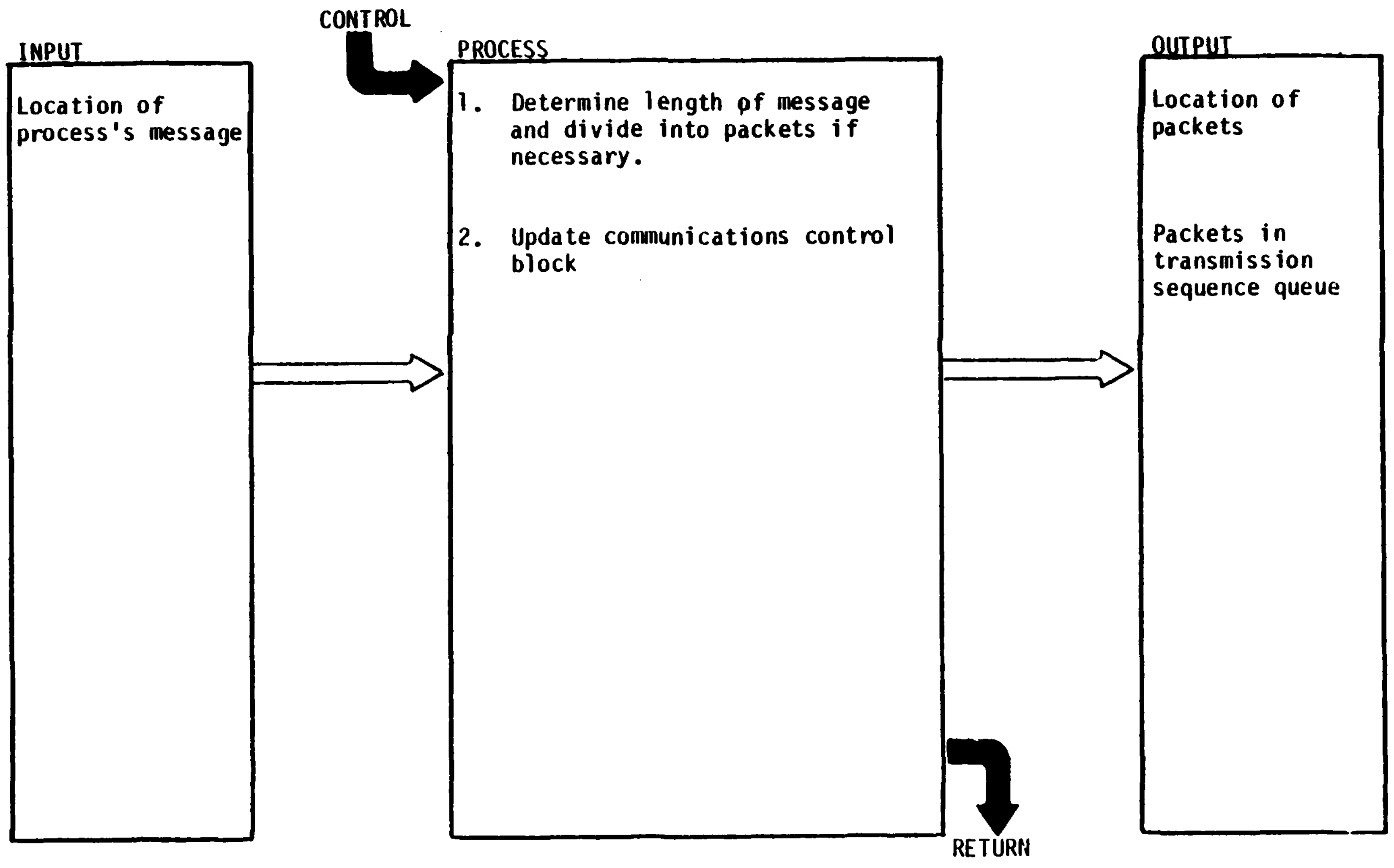
SYSTEM/PROGRAM Network Protocol DIAGRAM ID 2.0

NAME Send Message (Local Network)


INPUT
Destination address
Priority
Location
End of message indication
PROCESS
1. Break message into network size packets 2.1
2. Construct appropriate header for each packet 2.2
3. Access the channel 2.3
4. Send the packet 2.4
5. Receive control packet 2.5
OUTPUT
Partitioned message
Packets ready for transmission
Packet transmitted
Updated communication control information

SYSTEM/PROGRAM Network Protocol DIAGRAM ID 2.1

NAME Form Packets



INPUT

Location of process's message

CONTROL

PROCESS

1. Determine length of message and divide into packets if necessary.

2. Update communications control block

RETURN

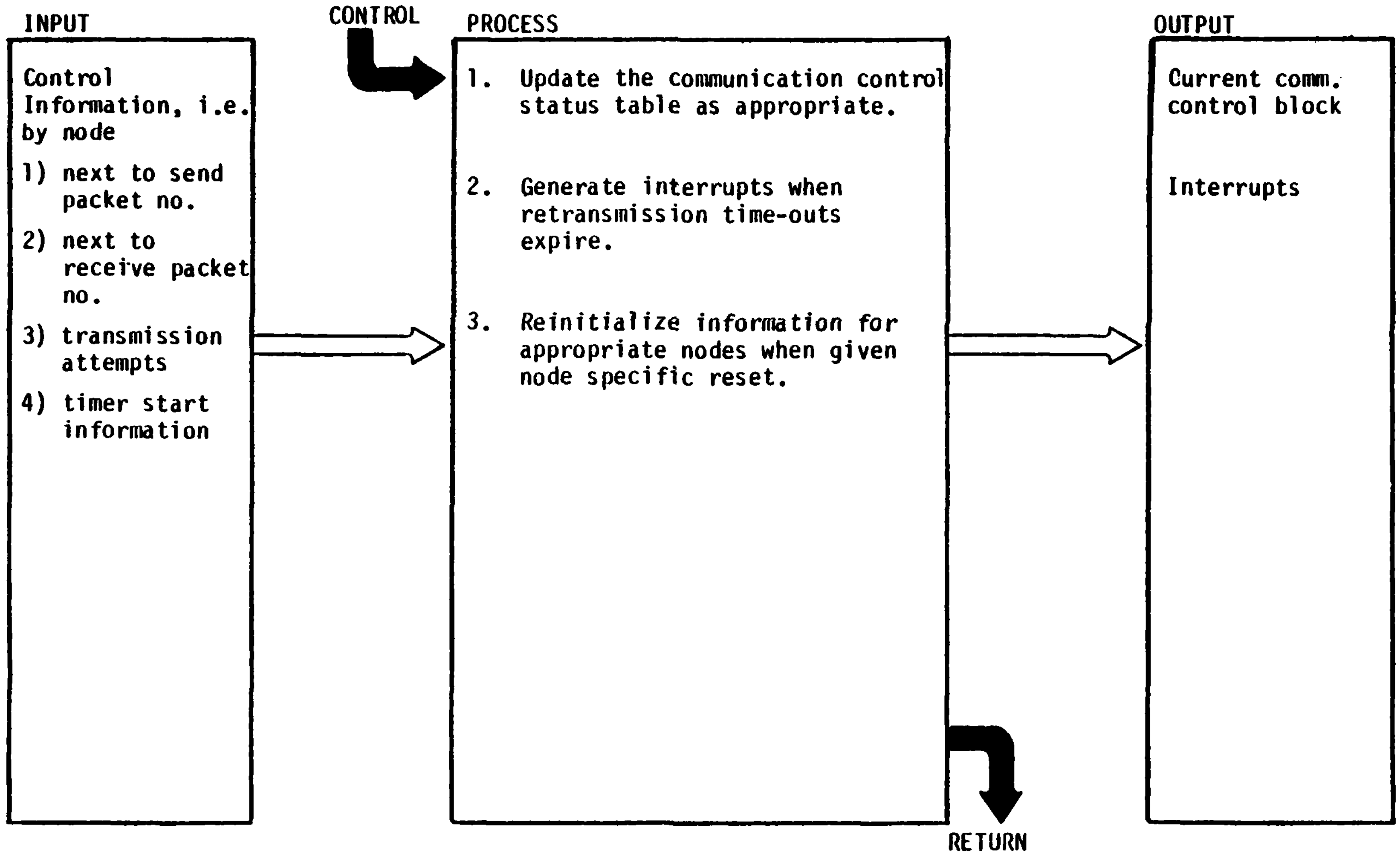
OUTPUT

Location of packets

Packets in transmission sequence queue

SYSTEM/PROGRAM Network Protocol DIAGRAM ID 2.1.1

NAME Update Communication Control Block



INPUT

Control Information, i.e. by node

1) next to send packet no.
2) next to receive packet no.
3) transmission attempts
4) timer start information

CONTROL

PROCESS

1. Update the communication control status table as appropriate.
2. Generate interrupts when retransmission time-outs expire.
3. Reinitialize information for appropriate nodes when given node specific reset.

RETURN

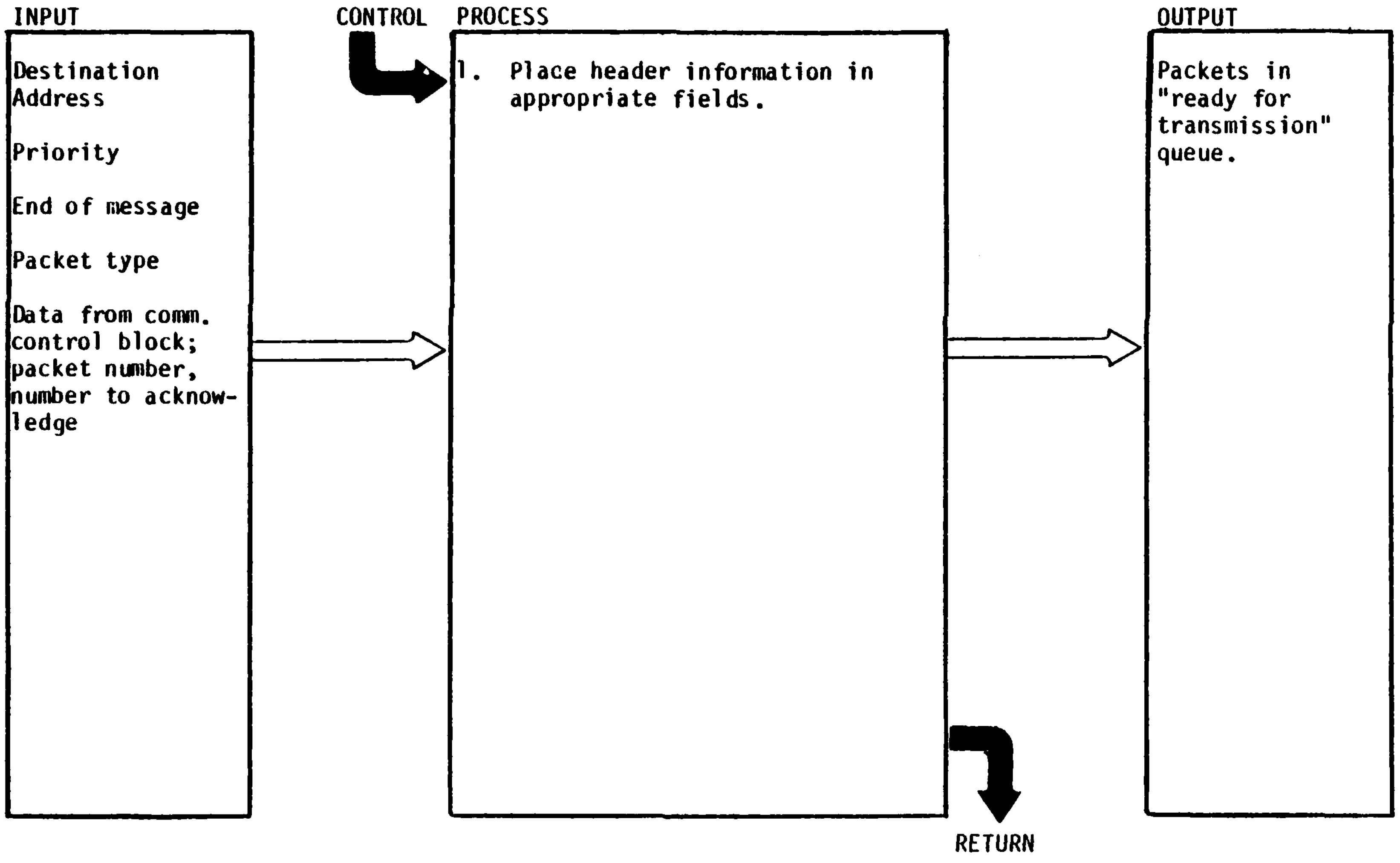
OUTPUT

Current comm. control block

Interrupts

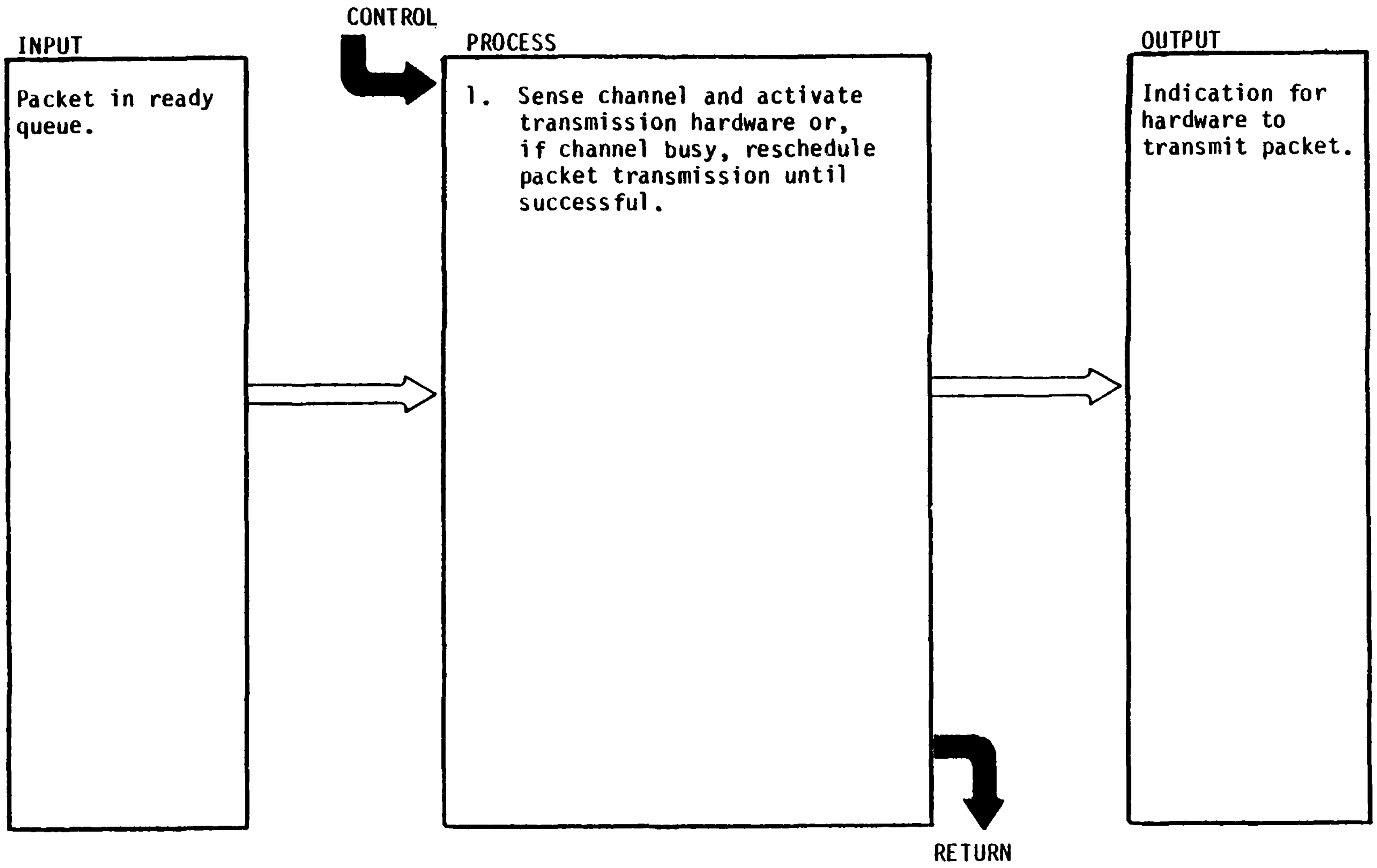
SYSTEM/PROGRAM Network Protocol DIAGRAM ID 2.2

NAME Build Header


INPUT
Destination Address
Priority
End of message
Packet type
Data from comm. control block; packet number, number to acknow-ledge
CONTROL
PROCESS
1. Place header information in appropriate fields.
OUTPUT
Packets in "ready for transmission" queue.
RETURN

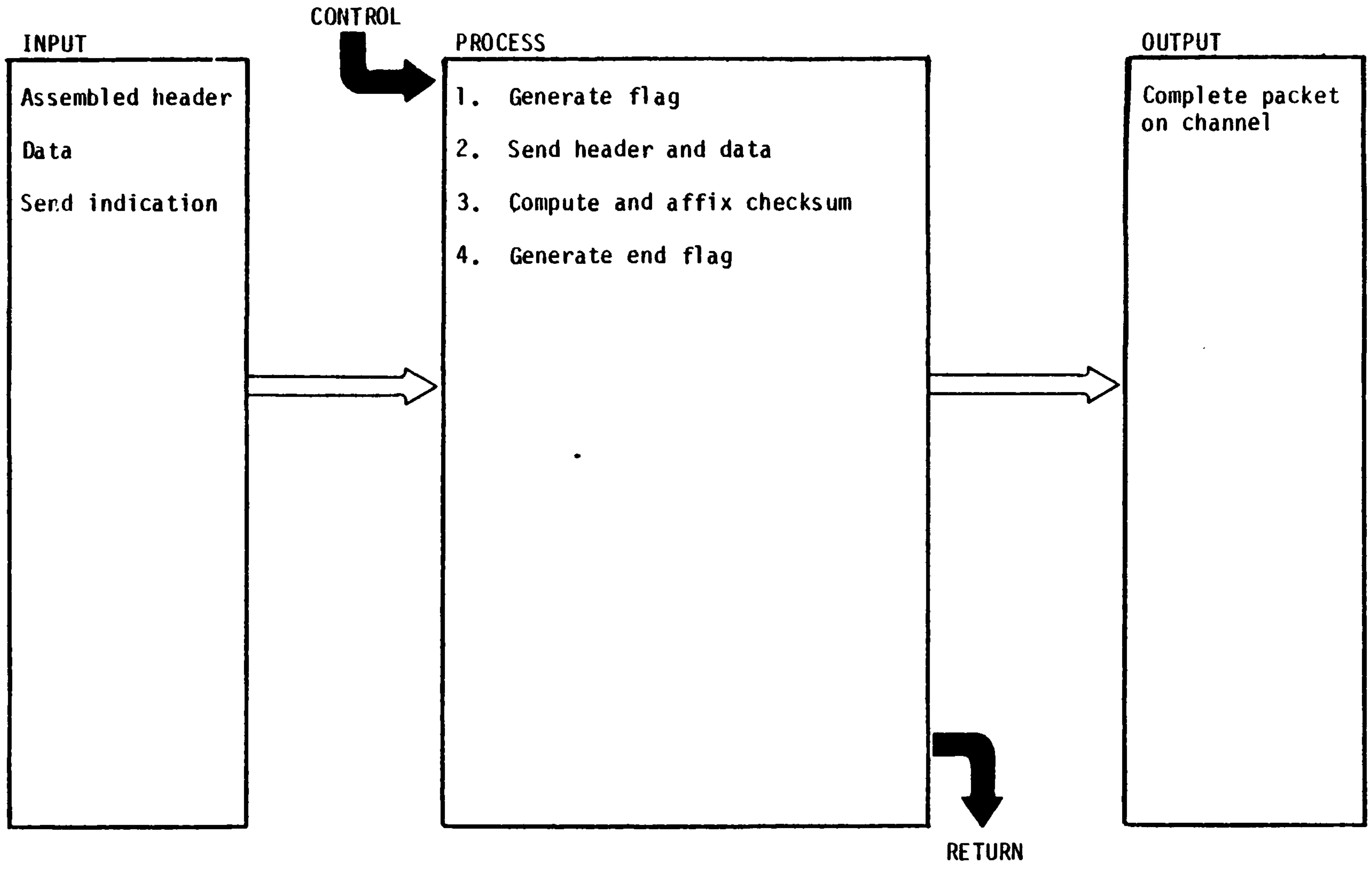
SYSTEM/PROGRAM Network Protocol DIAGRAM ID 2.3

NAME Access Channel


CONTROL
INPUT
Packet in ready queue.
PROCESS
1. Sense channel and activate transmission hardware or, if channel busy, reschedule packet transmission until successful.
OUTPUT
Indication for hardware to transmit packet.
RETURN

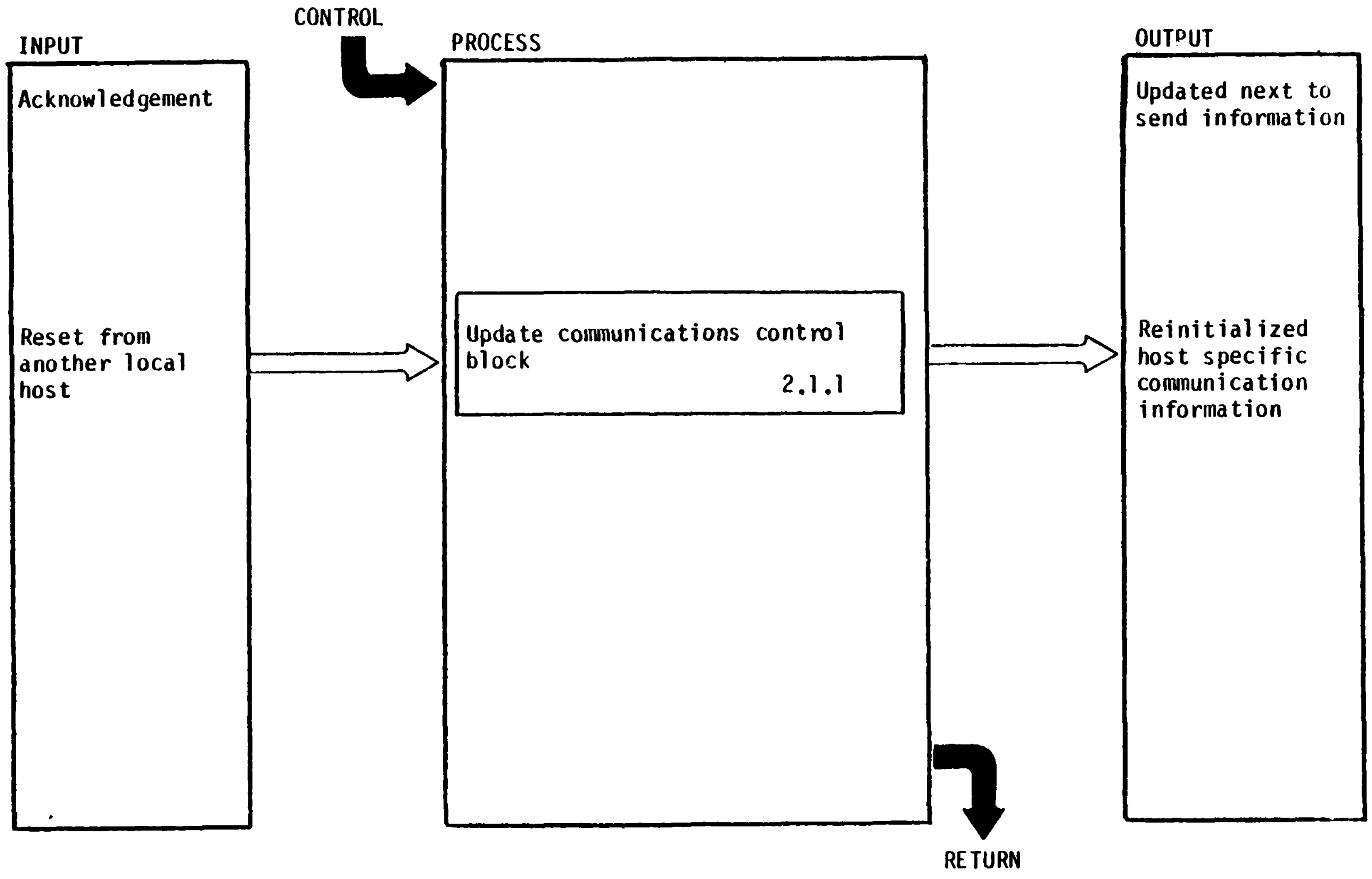
SYSTEM/PROGRAM Network Protocol DIAGRAM ID 2.4

NAME Send Packet



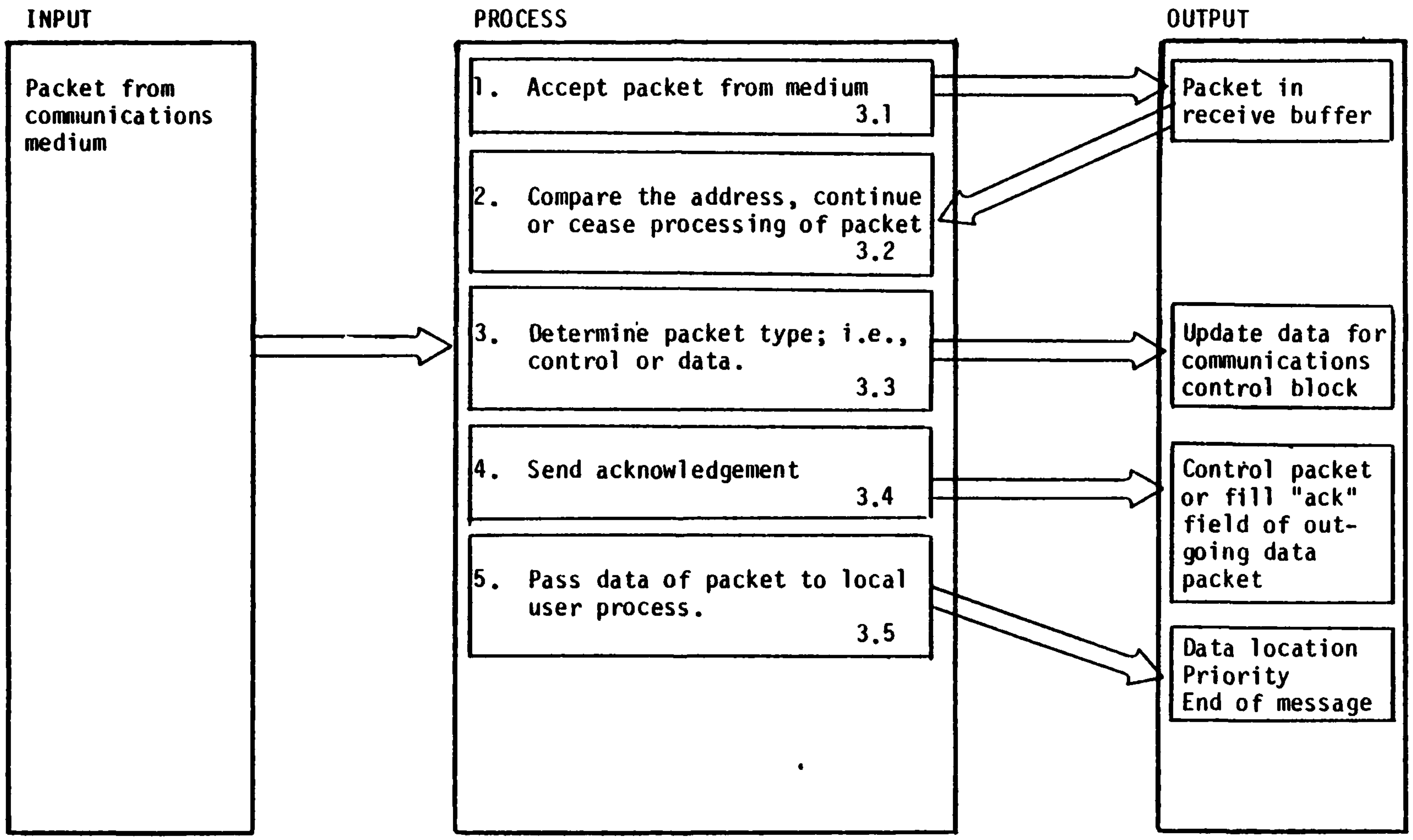
SYSTEM/PROGRAM Network Protocol DIAGRAM ID 2.5

NAME Receive Control Packet


INPUT
Acknowledgement
Reset from another local host
CONTROL
PROCESS
Update communications control block 2.1.1
RETURN
OUTPUT
Updated next to send information
Reinitialized host specific communication information

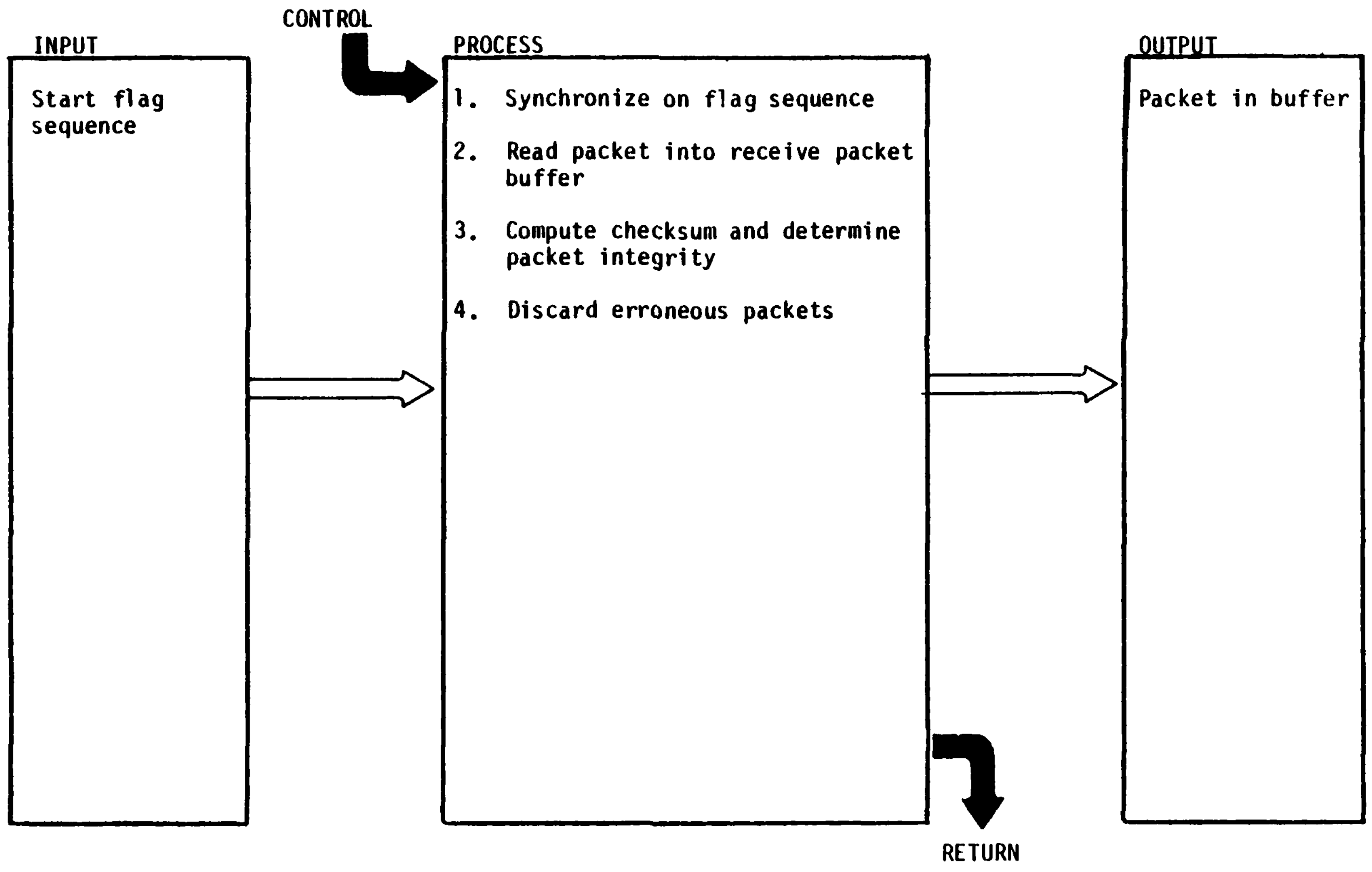
SYSTEM/PROGRAM Network Protocol DIAGRAM ID 3.0

NAME Receive Message


INPUT
PROCESS
OUTPUT
Packet from communications medium
1. Accept packet from medium 3.1
2. Compare the address, continue or cease processing of packet 3.2
3. Determine packet type; i.e., control or data. 3.3
4. Send acknowledgement 3.4
5. Pass data of packet to local user process. 3.5
Packet in receive buffer
Update data for communications control block
Control packet or fill "ack" field of outgoing data packet
Data location
Priority
End of message

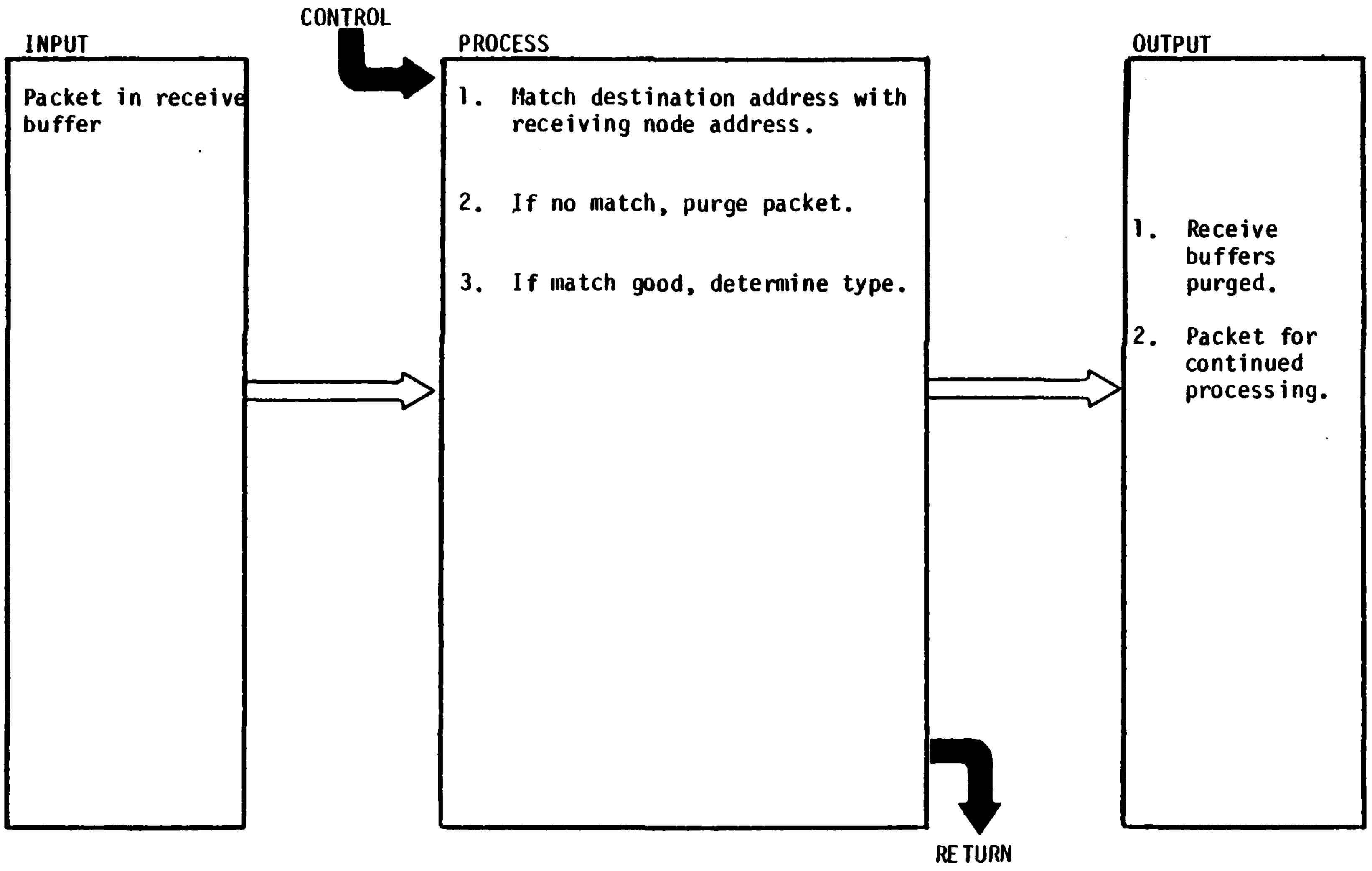
SYSTEM/PROGRAM Network Protocol DIAGRAM ID 3.1

NAME Accept Packet


CONTROL
INPUT
Start flag sequence
PROCESS
1. Synchronize on flag sequence
2. Read packet into receive packet buffer
3. Compute checksum and determine packet integrity
4. Discard erroneous packets
OUTPUT
Packet in buffer
RETURN

SYSTEM/PROGRAM Network Protocol DIAGRAM ID 3.2

NAME Read Address



INPUT

Packet in receive buffer

PROCESS

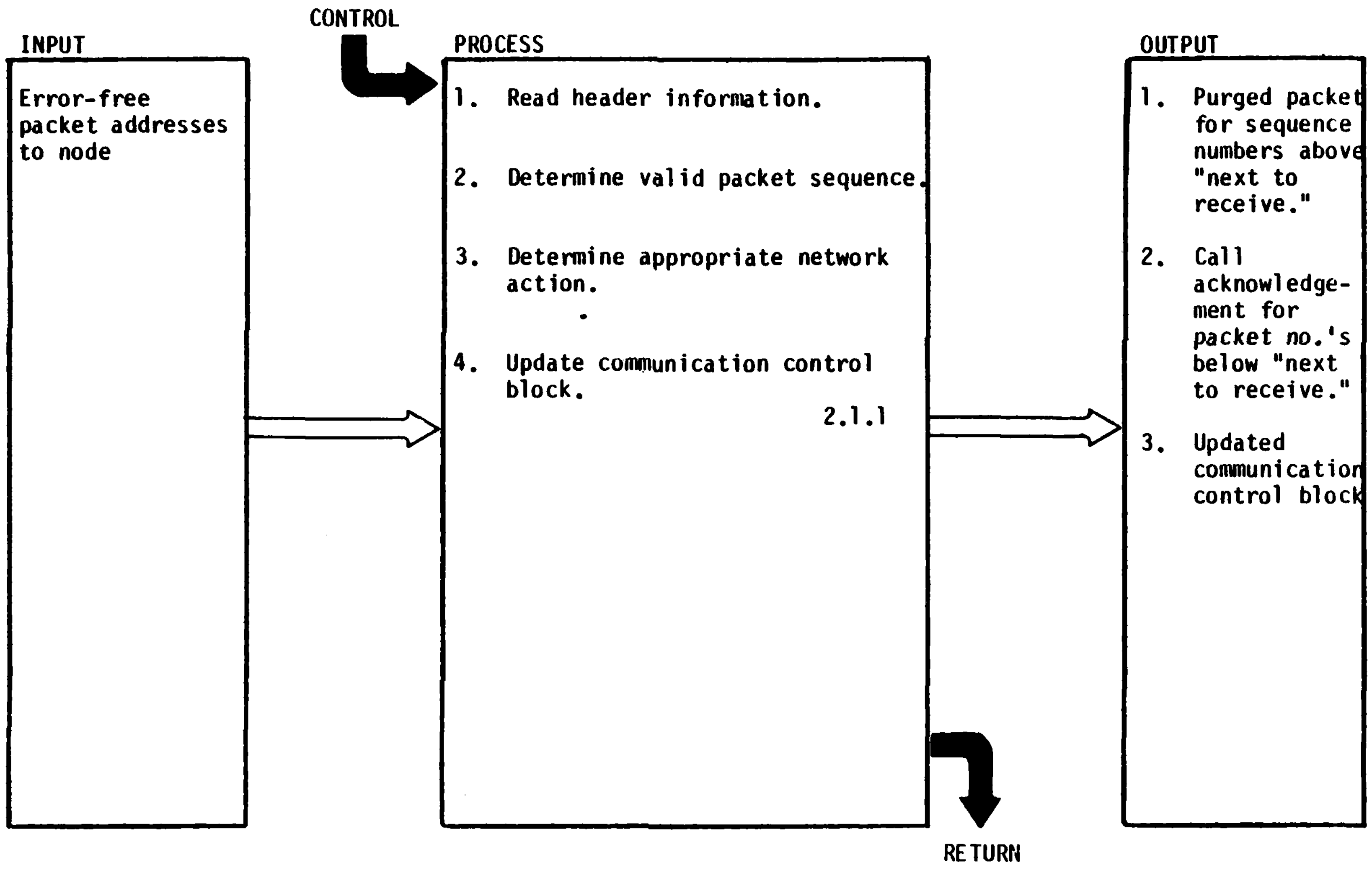
1. Match destination address with receiving node address.
2. If no match, purge packet.
3. If match good, determine type.

OUTPUT

1. Receive buffers purged.
2. Packet for continued processing.

SYSTEM/PROGRAM Network Protocol DIAGRAM ID 3.3

NAME Determine Type



INPUT

Error-free packet addresses to node

PROCESS

1. Read header information.
2. Determine valid packet sequence.
3. Determine appropriate network action.
4. Update communication control block.

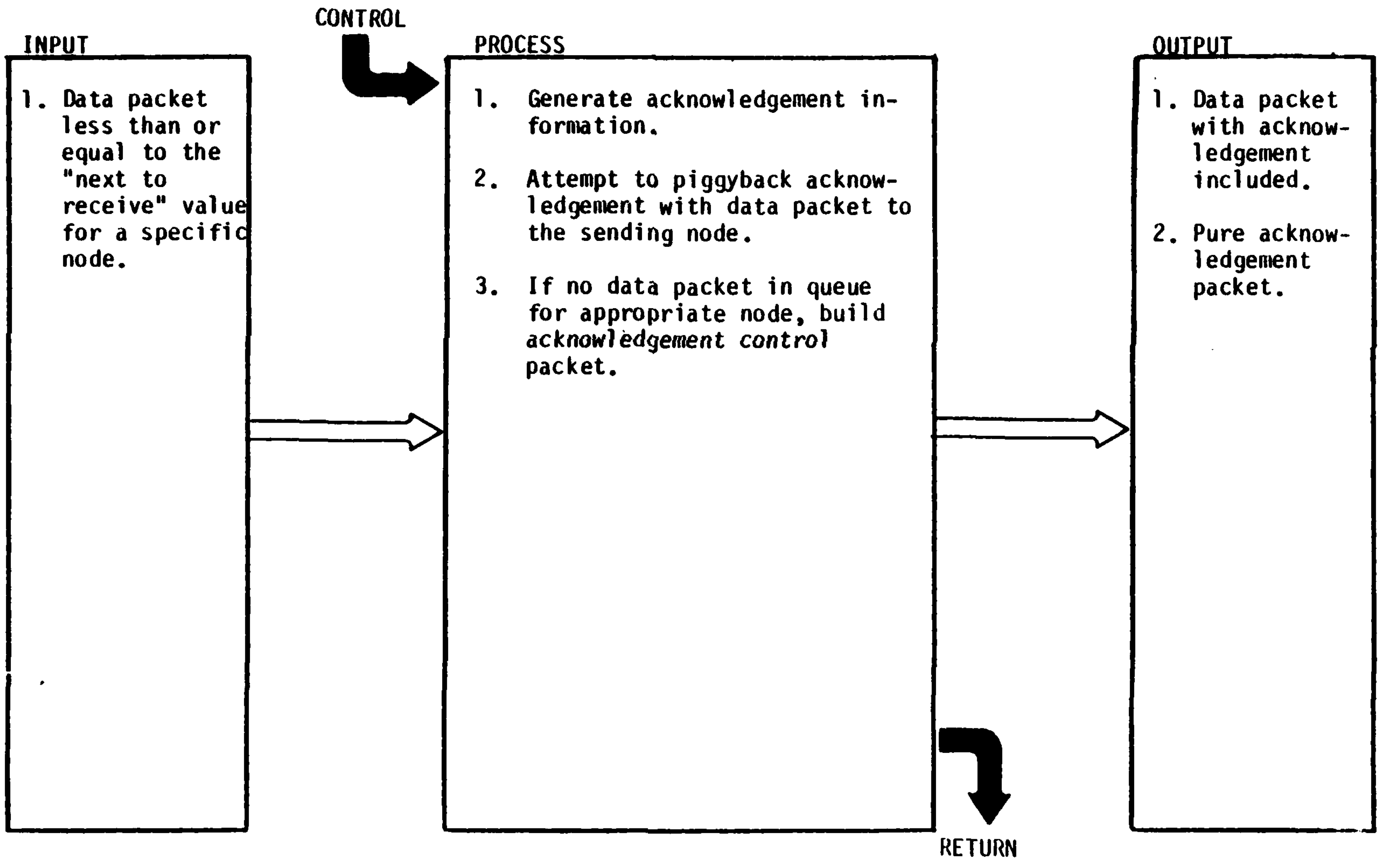
2.1.1

OUTPUT

1. Purged packet for sequence numbers above "next to receive."
2. Call acknowledgement for packet no.'s below "next to receive."
3. Updated communication control block

SYSTEM/PROGRAM Network Protocol DIAGRAM ID 3.4

NAME Send Acknowledgement



INPUT

1. Data packet less than or equal to the "next to receive" value for a specific node.

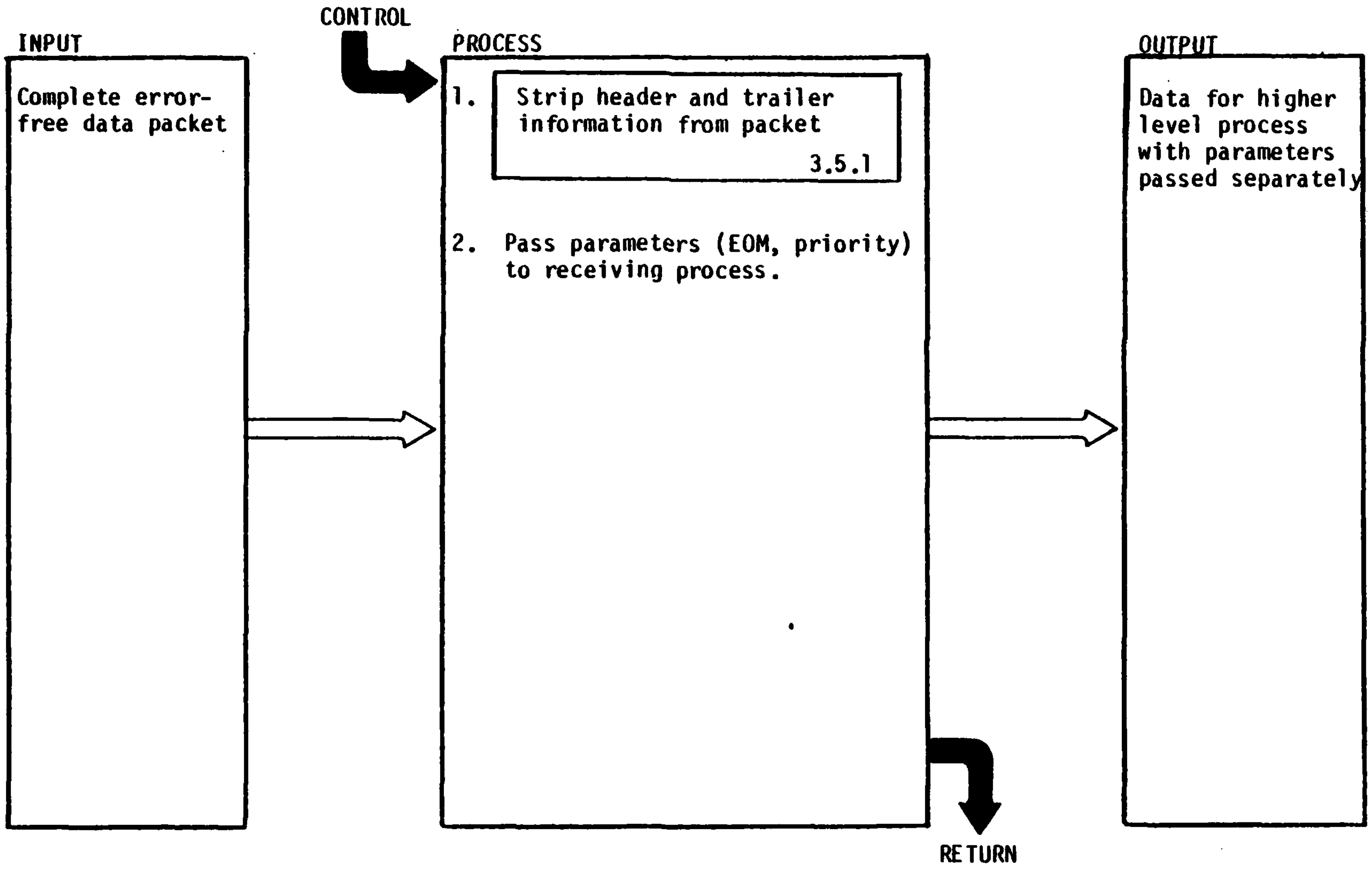
PROCESS

1. Generate acknowledgement information.
2. Attempt to piggyback acknowledgement with data packet to the sending node.
3. If no data packet in queue for appropriate node, build *acknowledgement control* packet.

OUTPUT

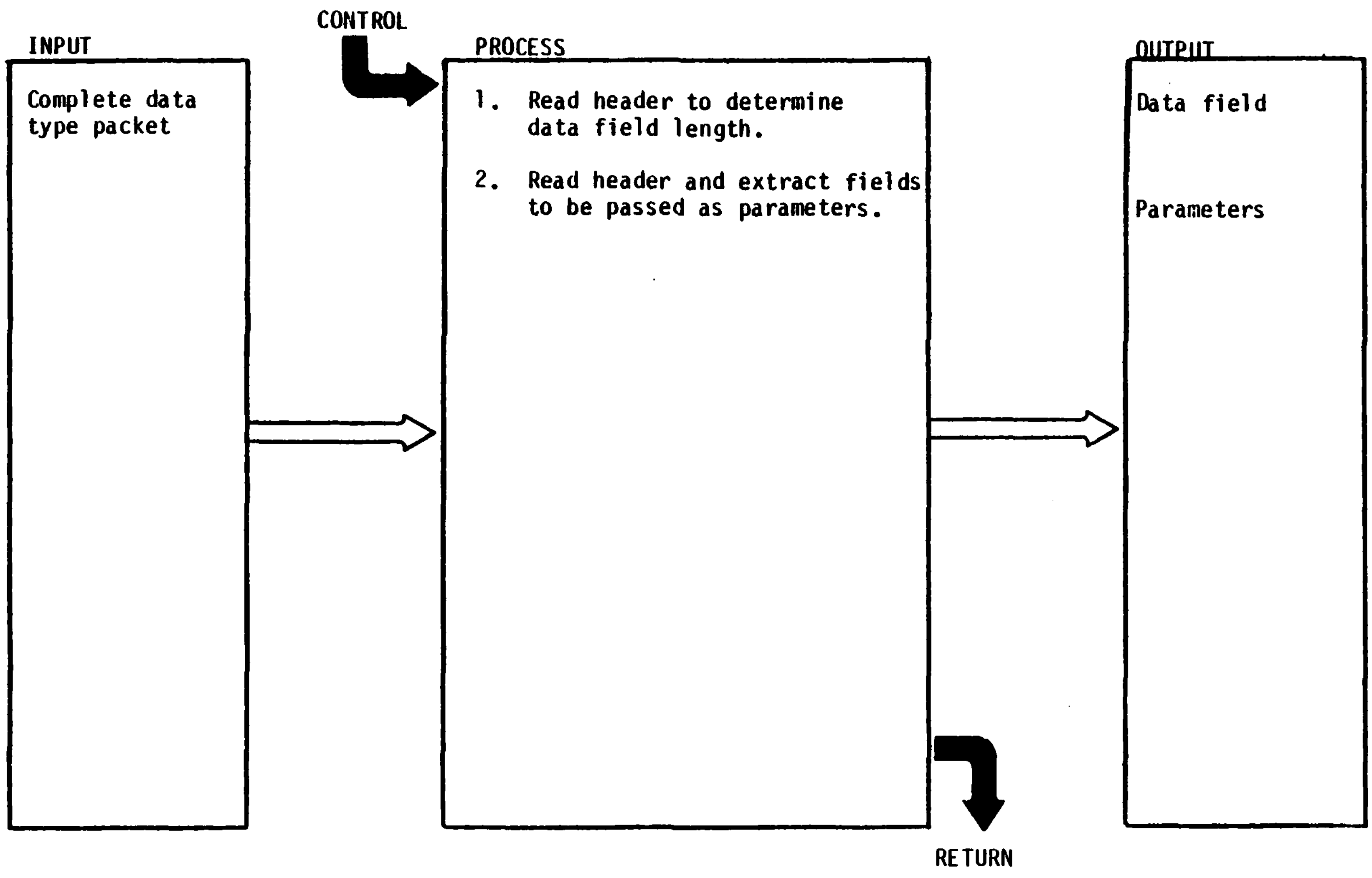
1. Data packet with acknowledgement included.
2. Pure acknowledgement packet.

SYSTEM/PROGRAM Network Protocol	DIAGRAM ID 3.5
NAME Pass Data to Higher Level Protocol	


CONTROL
INPUT
Complete error-free data packet
PROCESS
1. Strip header and trailer information from packet 3.5.1
2. Pass parameters (EOM, priority) to receiving process.
OUTPUT
Data for higher level process with parameters passed separately
RETURN

SYSTEM/PROGRAM Network Protocol DIAGRAM ID 3.5.1

NAME Strip Header Info



INPUT

Complete data type packet

CONTROL

PROCESS

1. Read header to determine data field length.
2. Read header and extract fields to be passed as parameters.

OUTPUT

Data field

Parameters

RETURN

APPENDIX B: WWMCS WIN Configuration

DTI Document Number 81045.C-WNFE.19

DRAFT

WNFE FUNCTIONAL DESCRIPTION

by

Edward R. Allen

Prepared for the
Defense Communications Agency
under contract
DCA100-81-C-0010
WWMCCS Network Front End Development

by

DIGITAL TECHNOLOGY INCORPORATED
302 E. John Street
Champaign, Illinois 61820

16 November 1981

Approved for Release: [illegible]
James P. Starks, Program Manager

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER 81045.C-WNFE.19	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) WNFE Functional Description		5. TYPE OF REPORT & PERIOD COVERED
		6. PERFORMING ORG. REPORT NUMBER 81045.C-WNFE.19
7. AUTHOR(s) Edward R. Allen		8. CONTRACT OR GRANT NUMBER(s) DCA100-81-C-0010
9. PERFORMING ORGANIZATION NAME AND ADDRESS Digital Technology Incorporated 302 East John Street Champaign, IL 61820		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS DCA/Command and Control Technical Ctr. WWMCCS ADP Directorate 11440 Isaac Newton Sq.N., Reston, VA		12. REPORT DATE 16 November 1981
		13. NUMBER OF PAGES 68
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release
Distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

No restriction distribution

18. SUPPLEMENTARY NOTES

None

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

WNFE	WWMCCS
NFE	WIN
Network Front End	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Digital Technology Incorporated is developing a Network Front End for the DoD World Wide Military Command and Control System (WWMCCS). This front end is called the WNFE. This document describes the function of the host-to-host front end and network protocols, and the woftware modules that implement these protocols. Also presented are descriptions of enhancements to (Continued on back)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. continued

the operating system software and peripheral equipment interface software.

CONTENTS

ILLUSTRATIONS

SECTION 2.

WNFE FUNCTIONAL DESCRIPTION

2.1 Introduction

The Worldwide Military Command and Control System (WWMCCS) serves the National Command Authorities and key military commanders across a broad spectrum of planning and operation activities. These activities can range from day-to-day crisis operations to conventional and nuclear war. This system involves 82 Honeywell 6000-series computers at 26 sites. The WWMCCS Intercomputer Network (WIN) provides a high-speed, high-capacity digital data communication capability among 20 of these sites.

The addition of network front end computers at selected sites is a part of the modernization of the WWMCCS Information System (WIS). A network front end (NFE) is a computer system interposed between a host computer and a computer network. The WNFE is the network front end computer for the WWMCCS Intercomputer Network (WIN). The WNFE removes much of the network interface software from the WWMCCS Honeywell H6000 host computers. This frees host storage and processor cycle time for other tasks. Terminals and other peripherals can be connected directly to the WNFE system, which further offloads some processing tasks from the host and provides an alternate access to the network.

The WNFE supports Department of Defense (DoD) network protocols and mediates communication channels between

a. Local host programs and remote host programs

b. Local host programs and remote terminals

c. Local terminals and remote host programs

d. WNFE terminals and local host programs

e. WNFE terminals and remote host programs

2.2 System Overview

This subsection presents an overview of the WNFE system. Following subsections briefly describe each of the WNFE modules. Detailed information on each protocol module is provided in the respective section of this manual.

2.2.1 WNFE Hardware. The WNFE is implemented on a Digital Equipment Corporation (DEC) PDP-11/70 computer. The standard hardware configuration includes

a. A PDP-11/70 Central Processing Unit with DL11 Console Interface and KW11-K Line Clock

b. An LA120 Console

c. 1.5 megabytes of error-correcting-code (ECC) metal oxide semiconductor (MOS) memory with battery back-up

d. Two DEC RL02 disk drives, used for swap storage, system software, and bulk storage for data such as audit trail logs

e. Two RL11 Disk Controllers

f. Two Associated Computer Consultants Local Host/Distant Host Controllers (LH-DH/11) used to interface the WNFE with the host computer and with the network packet switch, the Interface Message Processor (IMP)

g. Two DEC DV11-AA Communications Controllers with up to two DEC DV11-BA Synchronous Multiplexers each, used to interface the WNFE with Honeywell Video Information Projector (VIP) terminals. Each multiplexer has 8 lines.

h. A DEC KW11-P Programmable Clock, used for high resolution timing functions

i. Two DH11-AE 16-line Asynchronous Line Controller, used to interface the WNFE with Teletype (TTY) type terminals

j. A DEC M9312 Bootstrap with custom code, used to load the operating system from the system disk pack

k. A Custom Conversion Subsystem, used to convert RS-232C signals to Mil Spec 188C signals

l. A DEC H960-DH Cabinet with expansion drawer

m. A DEC H960-C Cabinet; cabinet only

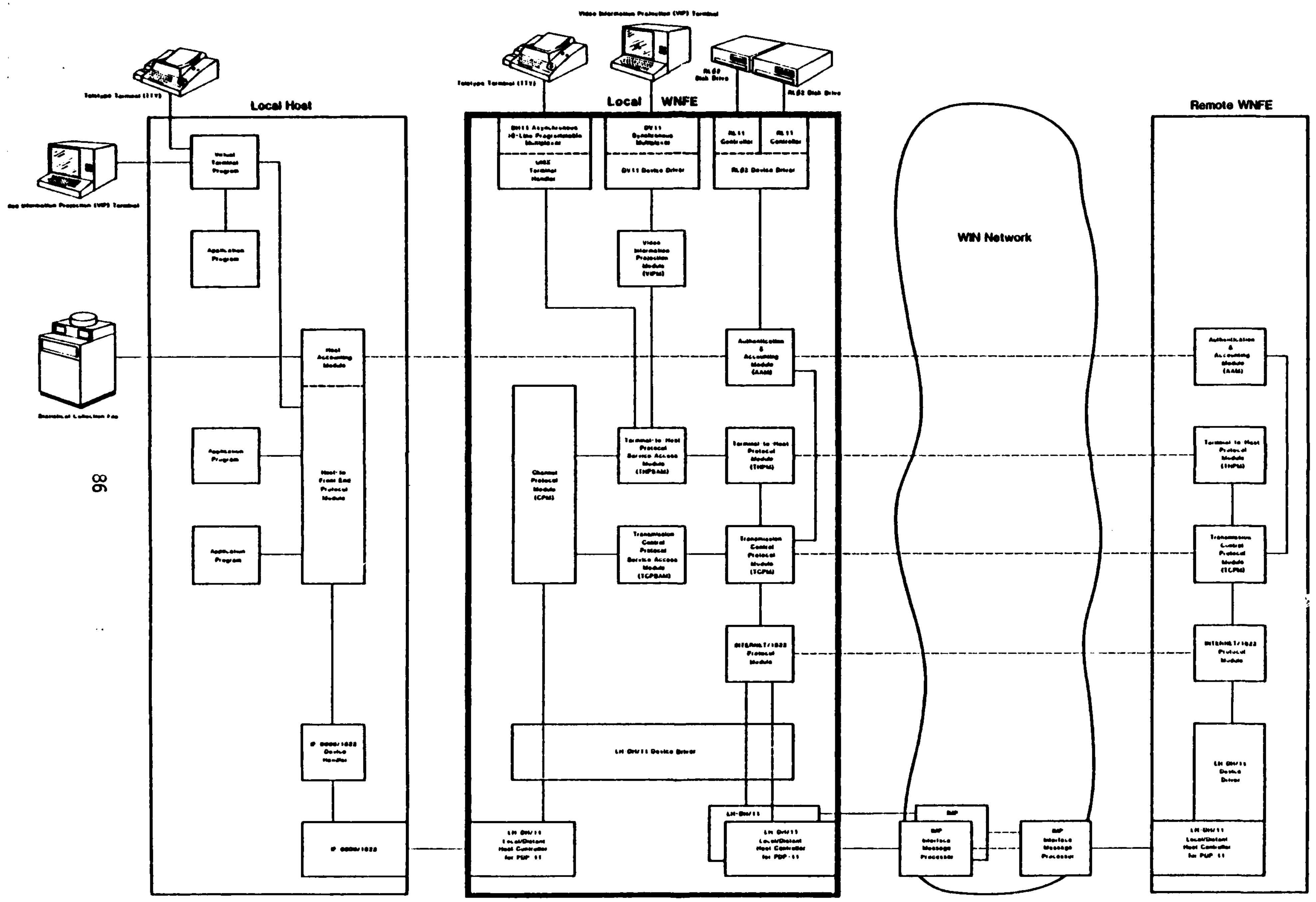

Local Host
Teletype Terminal (TTY)
Video Information Projection (VIP) Terminal
Statistical Collection File
Virtual Terminal Program
Application Program
Host Accounting Module
Host to Front End Protocol Module
IF 6000/1022 Device Handler
IF 6000/1022
Local WNFE
Teletype Terminal (TTY)
Video Information Projection (VIP) Terminal
RL02 Disk Drive
DH11 Asynchronous 16-Line Programmable Multiplexer
UNIX Terminal Handler
DV11 Synchronous Multiplexer
DV11 Device Driver
RL11 Controller
RL02 Device Driver
Video Information Projection Module (VIPM)
Authentication & Accounting Module (AAM)
Channel Protocol Module (CPM)
Terminal to Host Protocol Service Access Module (THPSAM)
Terminal to Host Protocol Module (THPM)
Transmission Control Protocol Service Access Module (TCPSAM)
Transmission Control Protocol Module (TCPM)
INTERNET/1822 Protocol Module
LH DH/11 Device Driver
LH DH/11
LH DH/11 Local/Distant Host Controller for PDP 11
WIN Network
IMP
IMP Interface Message Processor
Remote WNFE
Authentication & Accounting Module (AAM)
Terminal to Host Protocol Module (THPM)
Transmission Control Protocol Module (TCPM)
INTERNET/1822 Protocol Module
LH DH/11 Device Driver
LH DH/11 Local/Distant Host Controller for PDP 11

Figure 2-1. The WNFE System

The number and types of terminals and other devices will vary with site requirements.

2.2.2 WNFE Architecture. Figure 2-1 shows the relationships between a local host, a local WNFE, the WWMCCS Intercomputer Network (WIN) and a remote WNFE. The solid lines connecting the blocks represent the data paths that are actually traversed as data flows through the systems. These connections are referred to as real communications channels.

The dashed lines represent virtual data paths provided by the layered protocols. In a layered protocol system, each protocol layer provides a service to the next higher level and conceals the details of the lower levels from the user. For any given layer, there appears to be a direct communication channel between it and the corresponding layer in another system. These apparent channels are called virtual channels.

For example, messages exchanged between the Terminal-To-Host Protocol Module (THPM) in the local WNFE and a THPM in a remote WNFE must pass through several lower layers. However, the activities of the lower levels are invisible to both of the THPMs. The THPMs are able to exchange data as if there were a direct communication channel between them.

2.2.3 WNFE Software. The WNFE software consists of a modified Version 6 UNIX operating system, peripheral equipment device drivers and a device-specific protocol module, modules that implement host-to-front end and network protocols, and system control and monitoring modules.

2.2.3.1 UNIX Operating System Enhancements. Several modifications and enhancements to improve WNFE system performance have been made to the Version 6 UNIX operating system distributed by Western Electric Corporation. The primary enhancement is an interprocess communication mechanism called Attach I/O. The standard UNIX interprocess communication mechanism, called a pipe, is inadequate for network processing. Attach I/O is a low-overhead, nonblocking interprocess communication mechanism that is used for all WNFE interprocess communication.

2.2.3.2 Peripheral Equipment Interfaces. The WNFE provides device-specific protocol modules and device drivers to interface with

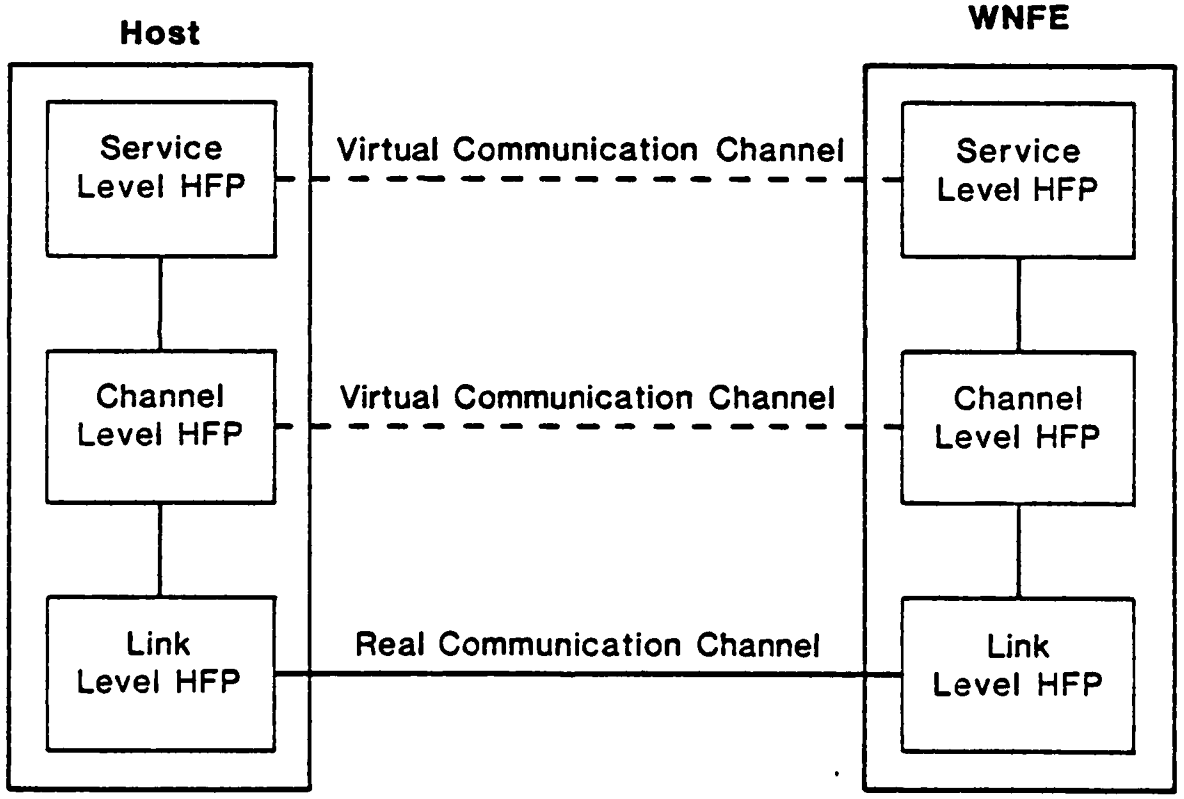
a. ASCII character mode TTY terminals

b. Honeywell Video Information Projector (VIP) terminals

c. The Digital Equipment RL02 Mass Storage Subsystem

The interface to ASCII character mode TTY terminals is provided by the DH11 Asynchronous 16-Line Programmable Multiplexer and the UNIX Terminal Device Driver that controls it. The interface to VIP terminals is provided by the Video Information Projection Module (VIPM) and the DV11 Synchronous Multiplexer Device Driver (DV11D).

2.2.3.3 Host-to-Front End Protocol. Local host processes and terminals communicate with the WNFE via the Host-to-Front End Protocol (HFP). The HFP provides host terminals and processes with access to the services provided by the WNFE. These services include the network services and services that provide access to WNFE terminals. The relationships between the levels of HFP are shown in Figure 2-2.



81-135

Figure 2-2. Host-to-Front End Protocols

The WNFE supports three levels of HFP:

a. The Link Level HFP governs communication over the hardware link connecting the local host and the WNFE. The Bolt, Beranek and Newman 1822 Protocol is used at this level. The protocol is implemented in a Local/Distant Host Controller and LH-DH/11 Device Driver. Both LH-DH/11s in the system are controlled by the same device driver.

b. The Channel Level HFP provides the virtual communication channels and flow control between the local host and the WNFE. The primary function of this level is the multiplexing and demultiplexing of HFP messages so that a number of virtual channels can share the bandwidth of the single host-to-WNFE physical channel. The Channel Level HFP uses the Link Level HFP as its communication medium. The Channel Level HFP is implemented in the Channel Protocol Module (CPM).

c. The Service Access Level HFP governs communication between local host programs and the services provided by the WNFE. This level uses the Channel Level HFP as its communication medium. The Service Access Level HFP converts HFP messages into the formats required by the modules providing the services. This level is implemented in two modules.

   1. The Terminal-to-Host Protocol Service Access Module (THPSAM) converts HFP messages coming from the host, and UNIX terminal handler data representations coming from WNFE terminals into THP messages, and vice versa. These conversions allow terminal users on the host or the WNFE to access the network services provided by the Terminal-to-Host Protocol Module (THPM). The THPSAM also converts HFP messages into UNIX terminal handler data representations and vice versa. These conversions provide WNFE terminal users with access to the local host.

   2. The Transmission Control Protocol Service Access Module (TCPSAM) provides host processes with access to the network connection services provided by the Transmission Control Protocol Module (TCPM).

2.2.3.4 Network Protocols. The WNFE communicates with remote WNFEs and remote hosts via standard DoD network protocols and an ARPANET host-to-IMP link protocol. The relationships between the modules that implement these protocols are shown in Figure 2-3.

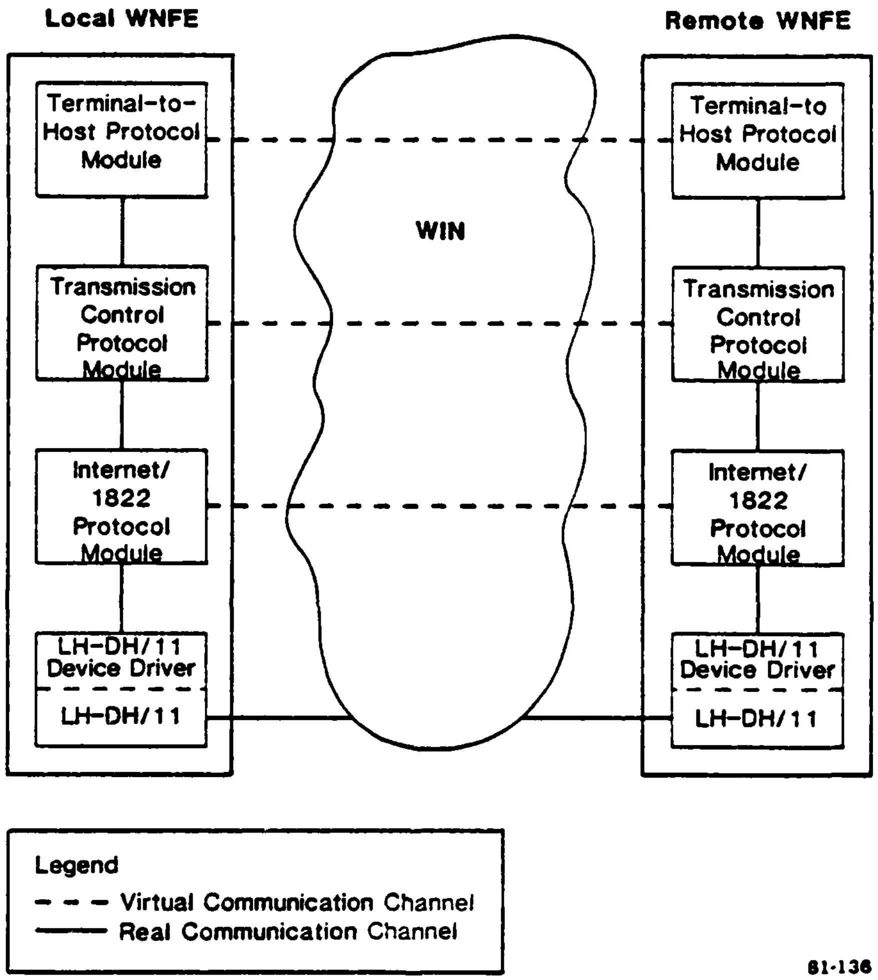

Local WNFE
Remote WNFE
Terminal-to-Host Protocol Module
Transmission Control Protocol Module
Internet/1822 Protocol Module
LH-DH/11 Device Driver
LH-DH/11
WIN
Terminal-to Host Protocol Module
Transmission Control Protocol Module
Internet/1822 Protocol Module
LH-DH/11 Device Driver
LH-DH/11
Legend
- - - Virtual Communication Channel
—— Real Communication Channel
81-136

Figure 2-3. Network Protocols

These protocols and the modules that implement them have the following functions:

a. The Terminal-to-Host Protocol (THP) provides network services to terminal users. It allows a variety of terminals and terminal-oriented programs to communicate over the network by compensating for different terminal characteristics and data representations. THP is implemented in the Terminal-to-Host Protocol Module (THPM) and the Terminal-to-Host Protocol Service Access Module (THPSAM).

b. The Transmission Control Protocol (TCP) provides connection-oriented, error-free, end-to-end data communications between the local WNFE and remote WNFEs or remote hosts. TCP is implemented in the Transmission Control Protocol Module (TCPM).

c. The Internet Protocol (IP) provides a simple datagram service over an interconnected system of networks. A datagram is a short message usually incorporated in a single packet. The WNFE implementation of IP includes provisions for multiple links to the network backbone, called <u>dual homing</u>. The Internet Protocol is implemented in the Internet Protocol - 1822 Protocol Module (IP1822M).

d. The Bolt, Beranek and Newman 1822 Protocol (BBN 1822) provides a reliable, high-speed link between the WNFE and the network packet switch, the Interface Message Processor (IMP). The implementation of the 1822 protocol is distributed. A portion of the protocol is implemented in the IP1822M. Most of the protocol is implemented in a Local Host/Distant Host Controller for PDP-11 (LH-DH/11) and the LH-DH/11 Device Driver. The device driver is a set of routines in the operating system.

2.2.3.5 Control and Monitoring Modules. The system control and monitoring functions for the WNFE are consolidated in two modules.

a. The Authentication and Accounting Module (AAM) starts all the other modules and acts as the control center for the WNFE system. Validation of all passwords, security codes, and access permissions is performed by the AAM. The AAM also controls the transfer of WNFE log files to the local host and the spawning of jobs on the host.

b. The Operator and Site Administrator Interface Module (OSAIM) provides a menu-driven interface for authorized operators of the WNFE system. The interface allows an operator, site administrator, or site security officer to select task modules that prompt for required information and perform control and monitor functions on the operating WNFE system.

2.2.4 WNFE Protocol Staging. Data flowing through the WNFE is handled by several modules. The routing and processing of data packages is called protocol staging. Protocol staging occurs along four routes:

a. Local host process to remote process route

b. Local host process to remote terminal and the local host terminal to remote process route

c. Local host process to WNFE terminal route

d. WNFE terminal to remote process route

Data is routed through the WNFE in the form of messages. A message consists of a block of data and one or more headers. A useful analogy is to think of the data portion of the message as a letter and a header portion of the message as an envelope. Using this analogy, each module then can be thought of as a postal station in a postal system. In general, each station (module) receives a letter (data) enclosed in one or more envelopes (headers). In other words, a station may receive a letter in an envelope which is contained in a second envelope, and so on.

The header contains routing and handling information, just as a postal envelope shows a destination, return address, postmark, and special handling instructions such as SPECIAL DELIVERY.

Each station (module) does one of two things when it receives a letter:

a. It encloses the letter in another envelope that contains the necessary routing information. This corresponds to a WNFE module creating and attaching a header to the message.

b. It removes the outer envelope and acts on the routing and handling information written on it. This corresponds to a WNFE module acting on the routing and handling information contained in the topmost header and then deleting it.

Notice that in this postal analogy each station need only be concerned with the outer envelope. Likewise, each module needs to examine the topmost header only.

The headers discussed above are specified by the protocol. Sometimes an additional header, called an <u>interprocess header</u>, that is not part of the protocol, is needed. It carries information required to implement the protocol. Taking the postal analogy a bit farther, the interprocess header can be likened to a note clipped to the outermost envelope. When interprocess headers are needed, the action taken by the modules is a little more complicated. Each stage first reads and removes the note, then adds or removes an envelope, and may clip a new note to the outermost envelope.

Figure 2-4 shows the staging along the route that would be followed by data being exchanged between a local host process and a remote host process. In the figure, the blocks at the bottom of the page represent the messages exchanged between the modules. Each of these blocks is divided to show the message format. The topmost section represents the outermost envelope or the note clipped to the outermost envelope.

A message received from the local host consists of a block of data and two headers: a TCP Service Level HFP Header and a Channel Level HFP Header. These headers were created and attached to the data by the Host-to-Front End Protocol module in the host.

The LH-DH/11 Device Driver passes the message unchanged to the CPM. The CPM examines the Channel Level HFP Header to determine how to handle the data. The CPM replaces the topmost header with an interprocess header and sends the message to the TCP Service Access Module. In general, interprocess headers contain information the neighboring module must know about the message to process it correctly. Some examples are message type, memory location of the message buffer, length of message, and control or status information.

The TCPSAM examines the interprocess header and the HFP header to determine how to process the message and deletes these two headers. It then adds a TCPSAM header, and passes the message to the TCPM. The TCPSAM header contains byte-size compensation information needed by the remote TCPSAM. This is required because the H6000 hosts use 9-bit bytes and the Transmission Control Protocol uses 8-bit bytes.

The TCPM examines and deletes the interprocess header, creates and adds the TCP header, and adds an interprocess header. It then sends the message to the IP module. The TCP header will be used by the receiving TCP to route the data within the receiving WNFE or host.

The IP module examines and deletes the interprocess header, creates the Internet header, and passes the message on to the 1822 Protocol Module.

The 1822 Protocol Module adds the Host-IMP header and calls on the LH-DH/11 Device Driver to transfer the message to the IMP.

In summary, the Host-to-Front End Protocol modules examine and delete HFP headers that were created in the local host. The network protocol modules create and add headers that will be examined by their counterparts in the remote WNFE system.

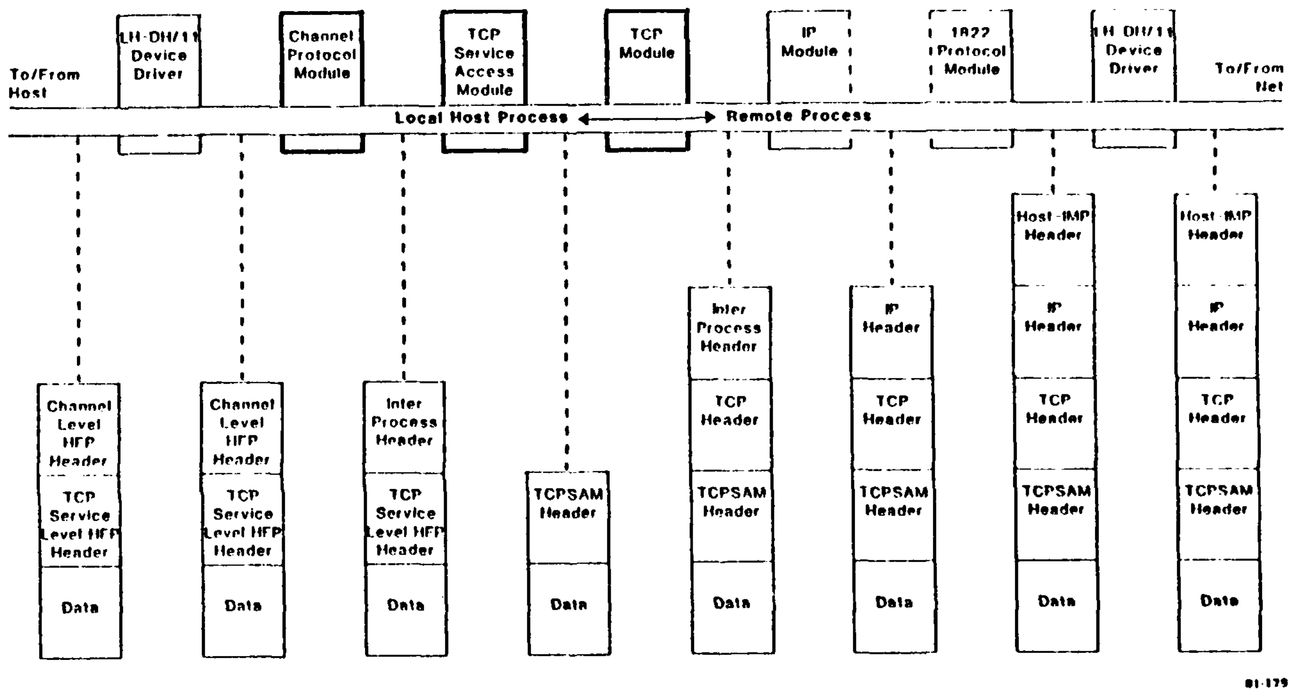


Figure 2-4. Local Process - Remote Process

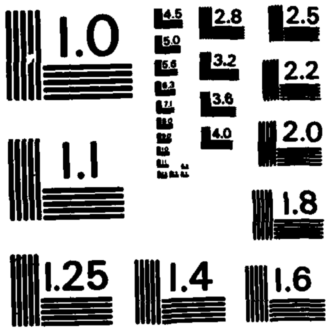
AD-A121 958 SUPPLY POINT LOGISTICS INTEGRATED COMMUNICATIONS ENVIRONMENT (SPLICE) LOC. (U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA K A INMAN ET AL. JUN 82 2/2

UNCLASSIFIED F/G 17/2.1 NL

END

FILMED

DTIC



When data is received from the network, the process described above is reversed. Modules that added protocol headers examine and delete them, and modules that deleted protocol headers build and add protocol headers to the data. The interprocess headers are used for the same purpose as before.

Figures 2-5, 2-6, and 2-7 show the other possible routes that data can take through the WNFE.

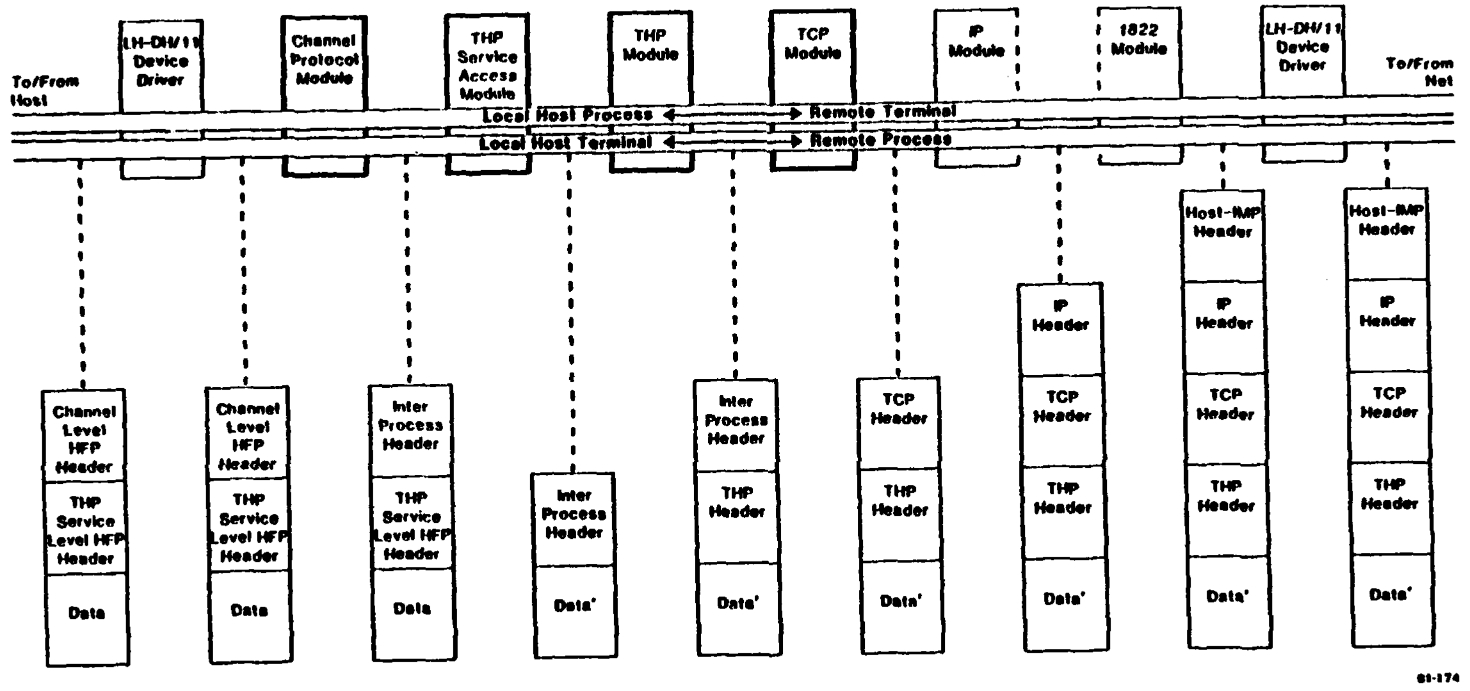


Figure 2-5. Local Process/Terminal - Remote Terminal/Process

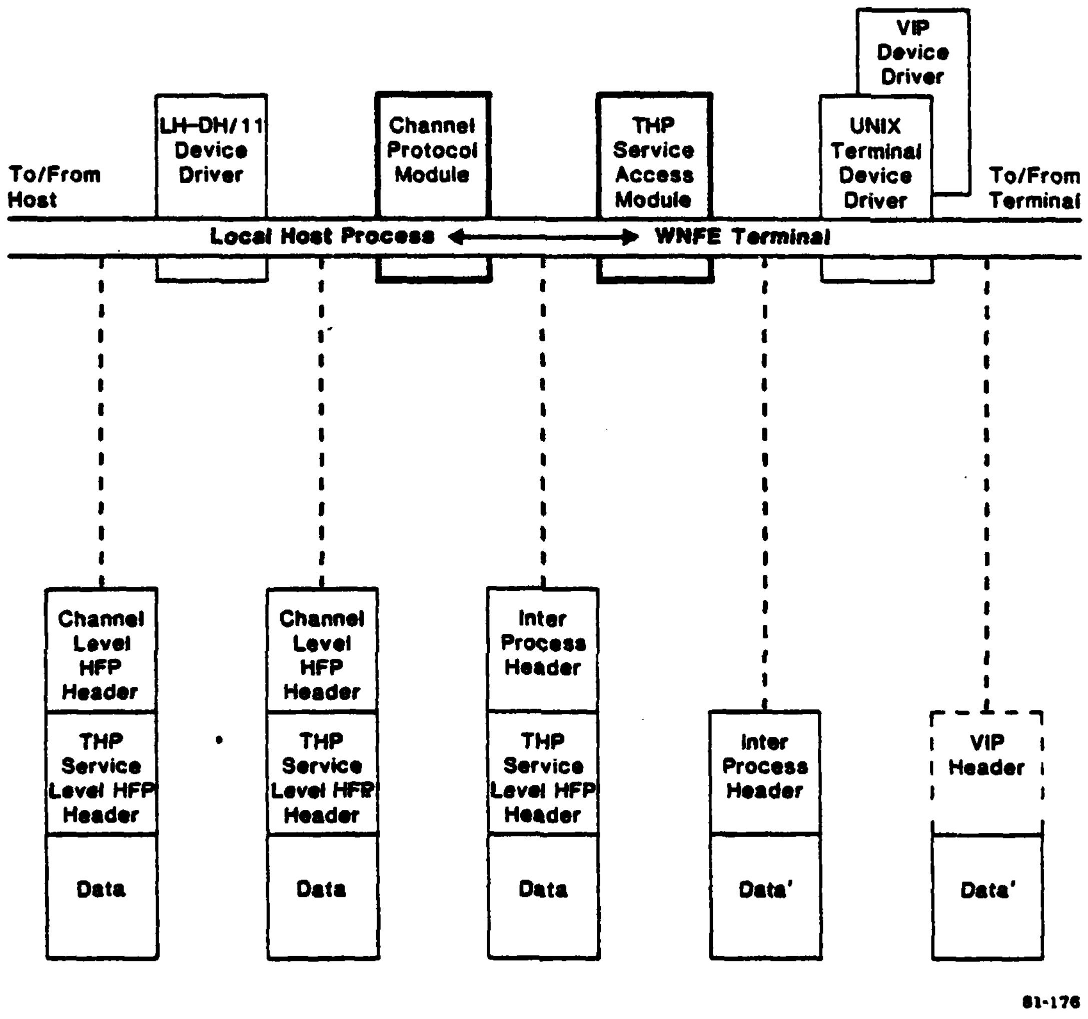


Figure 2-6. Local Process - WNFE Terminal

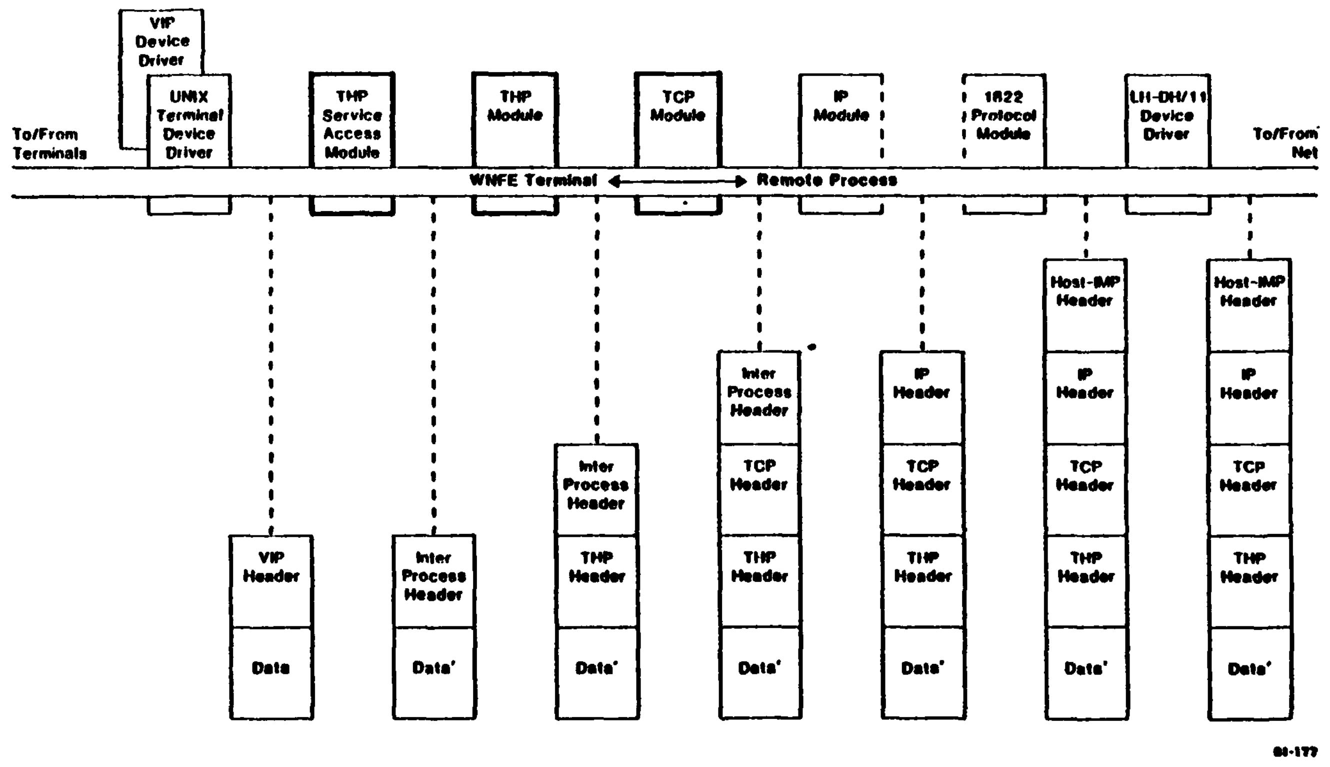


Figure 2-7. WNFE Terminal - Remote Process

2.3 Operating System Enhancements

The WNFE is based on a modified UNIX operating system. UNIX is a general purpose time-sharing operating system developed by Bell Laboratories for use on DEC PDP-11 computers. Several modifications were made to improve performance for network communications. Only the most important, Attach I/O, will be described here. A detailed description of Attach I/O and its implementation is provided in Section 3 of this manual.

An extension to standard UNIX I/O (input/output) provides a pipe mechanism for interprocess communications (IPC). With pipes, each time a process initiates an I/O or IPC operation, the execution of the process is suspended (blocked) until the operation is complete. This makes the mechanism slow and restricts a processes to only one I/O operation at a time.

Attach I/O was developed as a low overhead, non-blocking replacement for UNIX pipes. Attach I/O allows one process, called the responding process, to attach itself to the address space of another process, called the requesting process. This permits the responding process to access data directly in the memory address space of the requesting process. Thus, data copying can be avoided. Attach I/O supports both blocking and non-blocking operations. Using Attach I/O, a process is able to initiate multiple I/O or IPC operations without waiting for the first to complete, a necessary requirement for network operations.

The following paragraphs briefly compare Standard UNIX I/O and WNFE Attach I/O.

2.3.1 Standard UNIX I/O.

The UNIX file system is hierarchically structured. Each node in the file system represents a directory, a disk file, or a peripheral device driver such as a terminal handler.

A file is manipulated with four standard I/O primitives. The system calls are read, write, open, and close. An application level process requests an I/O function using one of the I/O primitives. The UNIX I/O system calls the corresponding device driver routine. The device driver routine then performs the I/O function. This is shown in Figure 2-8.

Some devices can be manipulated by special primitives such as the gtty (get TTY) and stty (set TTY) system calls used to examine and set terminal parameters.

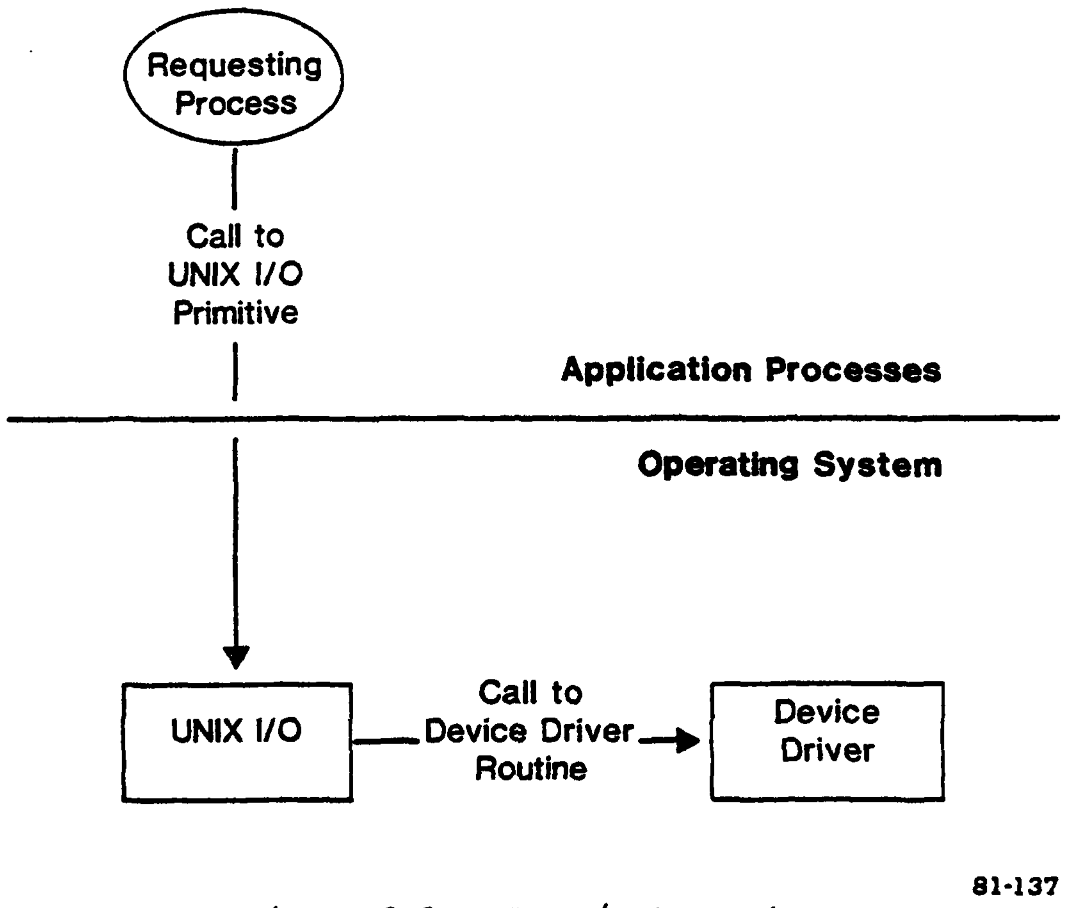


Figure 2-8. UNIX I/O Operation

Each peripheral device (e.g., line printer, terminal) has a set of routines in the operating system that implement the basic I/O primitive operations for the device. There is one routine for each primitive (e.g., the terminal read routine implements the read primitive operation for terminals). This set of routines is called a UNIX device driver.

2.3.2 Attach I/O. Attach I/O is an extension of the standard UNIX file system. A new file type has been added to UNIX: the attach file type. Attach files are created by application level processes. Attach files are opened, closed, read, and written by other processes, just like any other file. The operating system directs all I/O operations on an attach file to the process that created it. Through Attach I/O, an application level process can respond to I/O requests in a manner similar to a device driver.

For Attach I/O interprocess communication, one process is always the requesting process and the other is the responding process. The responding process creates the attach file. The requesting process then uses I/O primitives to communicate with the responding process. Two additional I/O system calls are unique to Attach I/O, **iospcl** and **ioreply**. A call to **iospcl** is used to transfer control information between modules. The responding process uses the Attach I/O system call **ioreply** to communicate with the requesting process so as to respond to the I/O requests.

When a requesting process calls the primitives, Attach I/O translates these calls into corresponding Attach I/O request messages, which are delivered to the responding process. For example, an open call results in an Attach I/O Open request being delivered. Only one reply primitive is required. The Attach I/O system associates the ioreply call with the corresponding request previously received and delivers an Attach I/O response message to the requesting process.

2.3.2.1 I/O Messages. There is a device driver for attach files. Unlike other device drivers, the Attach I/O device driver does not manage a peripheral device. Instead, it calls operating system routines that create I/O messages and passes the messages to the responding process (see Figure 2-9). I/O messages can be either I/O requests to the responding process or I/O replies from the responding process. An I/O message contains a function code that identifies the message type (e.g., Open request, Write reply) and parameters that identify a data buffer. The data buffer contains data to be transferred.

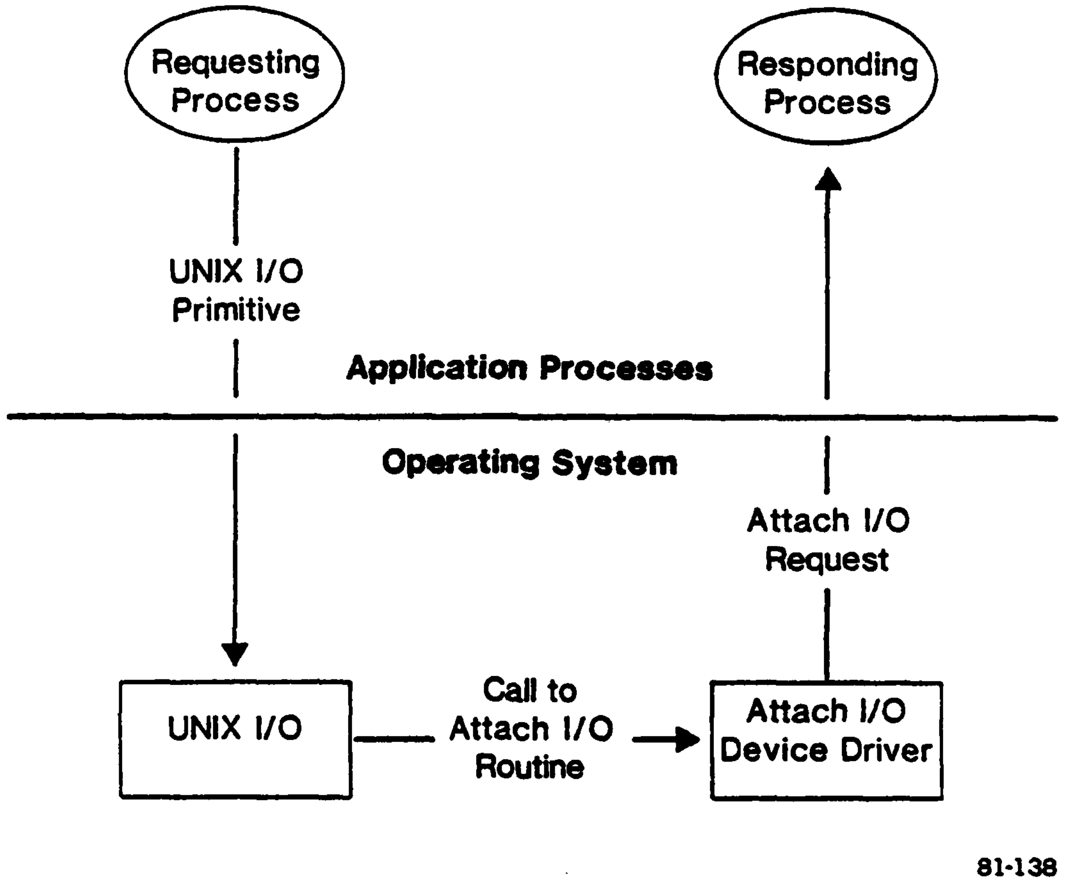


Figure 2-9. Attach I/O Operation

2.3.2.2 Buffer Access. A responding process uses system calls to access the I/O data buffer of a requesting process. The operating system uses the PDP-11/70 memory mapping hardware to allow both processes to access a common segment of memory.

On the PDP-11/70, the data space of a process is divided into segments. Corresponding to each segment is a hardware relocation register that points to the location of the segment in physical memory. The operating system sets one of the responding process's relocation registers to point to the requesting process's segment containing the desired data buffer. As a result, the data segment, and hence the buffer, can be directly accessed by both processes.

2.3.2.3 Blocking and Non-blocking Attach I/O. Attach files can be opened in either a blocking or non-blocking mode. When an I/O operation is requested in the blocking mode, control is not returned to the requesting process until after an I/O reply has been received from the responding process.

In the non-blocking mode, control is returned to the requesting process without waiting for an I/O reply from the responding process. At a later time the requesting process can call the operating system to get the I/O reply. Thus, through non-blocking Attach I/O, a requesting process is never blocked waiting for data on one attach file while data is available on another.

2.3.2.4 Attach I/O Operation. The sequence of events for a typical use of Attach I/O is as follows:

1. The responding process creates an Attach I/O file.

2. The requesting process sends an Open request to the responding process.

3. The responding process replies with an Open reply.

   At this point an <u>Attach I/O stream</u> is said to exist between the two processes. Data can be passed from the requesting process to the responding process in buffers described by Write requests. Data can flow in the opposite direction in buffers described by Read replies. The Read reply buffers must have been given to the responding process by the requesting process.

4. The Attach I/O file (and the stream) is closed when the requesting process sends a Close request and receives a Close reply.

2.4 Device Drivers and Device Protocol Modules

Several device drivers and a device-specific protocol module have been added to UNIX to support hardware unique to the WNFE. The device drivers are for

1. The RL11 controllers, which control the RLØ2 disk drives

2. The DV11 Synchronous Multiplexer (DV11), which is the synchronous line driver used to interface VIP terminals to the WNFE

3. The Local/Distant Host Controllers, which implement the 1822 line protocol and are used to interface the WNFE to the local host and to the IMP

The device-specific module is the Video Information Projection Module (VIPM), which implements the VIP communication protocol.

2.4.1 WNFE RLØ2 Device Driver. The WNFE disk storage subsystem is composed of

a. Two RL11 controllers

b. Two RLØ2 drives

c. RLØ2K disk cartridges (one per drive)

The WNFE system is configured with dual drive units and dual controllers to provide the reliability required for the WIN. A single device driver supports both RL11 controllers and both drive units. The redundancy of drives and controllers allows the system to operate, in a performance degraded mode, with only one drive unit and one controller operational.

The RLØ2K disk cartridge is a top-loading, removable disk cartridge containing a single platter. The cartridge can store 1Ø.4 megabytes of formatted data. There are 512 tracks on each platter surface. Each track is divided into 4Ø sectors with each sector containing 256 bytes of data.

The RL11 controller interfaces to software via four registers connnected to the I/O bus. The RL11 accepts commands to report subsystem status, control drive unit operation, and perform direct memory access (DMA) transfers between main memory and the drive units.

The RLØ2D provides routines to handle the I/O primitive operations for the disk system. Separate routines perform core dumps and load the bootstrap programs that initially load the operating system from the disk.

The RLØ2D is loaded as part of the UNIX operating system (Kernel). It is used to maintain the file system stored on the disk and to swap executing programs into and out of main memory. The driver is also used by the operating system on behalf of executing programs to read and write disk files, such as the authentication and log files.

Bootstrap loading of the UNIX operating system is initiated by loading a short, first stage bootstrap routine from the first sector of the disk on drive unit Ø. This routine in turn reads a larger, second stage bootstrap routine, which loads in the operating system and gives the operating system control.

2.4.2 DV11 Device Driver. The DV11 Device Driver (DV11D) is loaded as part of the UNIX operating system. It provides routines that handle I/O primitives for terminal devices that use synchronous line protocols. In the WNFE, the DV11D serves as the interface between the Video Information Projection Module (VIPM) and the DV11 Synchronous Multiplexer. The DV11 interfaces with VIP terminals connected directly to the WNFE.

The DV11 is a microprocessor-based device that handles synchronous data transmission and reception via direct memory access (DMA). It supports up to 16 synchronous serial data lines. Under direction of the DV11D, the DV11 prepares lines for use, moves data to and from PDP-11/70 memory, monitors and reports error conditions, calculates error detecting codes, and implements several communications protocols. Since the DV11 does much of the actual work involved in I/O operations, the PDP-11 is freed for more complex tasks.

Communication between the VIPM and the DV11D is via Attach I/O. The VIPM uses the I/O primitives **read** and **write** to receive data from and send data to VIP terminals. These calls result in requests being delivered to the DV11D. These requests describe VIPM data buffers that are the sources or destinations for data. The DV11 processes the requests and directs the DV11 to perform the data transfers.

The DV11D controls the DV11 by reading from and writing to the DV11 status and control registers. Data to be transferred is read directly from the VIPM output buffers and data received is written directly to the VIPM input buffers. As each data transfer completes, the DV11 notifies the DV11D, which in turn, notifies the VIPM by replying to the corresponding request.

2.4.3 LH-DH/11 Device Driver. The LH-DH/11 Device Driver (LHDHD) provides the software interface to the Local/Distant Host Controller (LH-DH/11) manufactured by Associated Computer Consultants. The LH-DH/11 implements most of the low level functions of the BBN 1822 protocol. Two LH-DH/11s are used in the system; one for the host-to-WNFE link and one for the WNFE-to-IMP link. The LHDHD is purely a transport mechanism. It does not process the content of the messages. Communication between the LHDHD and a using process is via Attach I/O.

2.4.3.1 Host-to-Front End Link. In the host/WNFE configuration (see Figure 2-2 page 2-4), the LH-DH/11 communicates with the IF-6000/1822, its counterpart in the H6000 host, via the Bolt, Beranek and Newman 1822 protocol. The IF-6000/1822, also manufactured by Associated Computer Consultants, is often incorrectly referred to as the ABSI (Asynchronous Bit Serial Interface).

2.4.3.2 WNFE-to-IMP Link. The WNFE-to-IMP link uses the same device driver as the Host-to-Front End link, but a separate LH-DH/11. The IMP is a BBN C30 computer that has an 1822 protocol interface as standard equipment.

2.4.3.3 Link Operation. The following description of operation of the LH-DH/11 and device driver applies to both links. The LH-DH/11 is composed of two DMA controllers that attach to the DEC PDP-11 I/O bus, thus providing full-duplex communication.

The 1822 protocol was designed to provide asynchronous bit-serial communication between a host computer and an ARPANET Interface Message Processor (IMP), but it can be used to interface any two computers.

The BBN 1822 protocol is full duplex and symmetrical. The protocol specifies the use of ten signal lines. Viewed from either side of the communication link, four signal lines are used for input, and four are used for output. The remaining two lines are used to indicate the operational status of the machines at either end of the link. Of the four lines used for input (or output), one line carries the data bits, two lines are used for handshaking on a bit-by-bit basis, and one line indicates transfer of the last bit of a message.

The handshaking signals allow either machine to limit the data transfer rate. The Last-bit indicator informs the receiver when the message is complete without the receiver having to examine the data stream. The machine at each end of the link controls a Master Ready Relay that can be monitored at the other end. This allows both machines to know if the other is up or down without any message exchange.

When the link channel is opened by the user process, the LHDHD prepares the LH-DH/11 to receive and transmit data. When the link is closed by the user process, the LHDHD disables the interface and returns any outstanding data transfer requests it may have to the user.

The user uses read and write system calls to receive and send data over the link. As a result of a system call, the LHDHD receives a request that describes the user's input or output buffer and a count of the bytes to be transferred. For each request, the driver sets up the device via the control registers and signals the device to start the transfer. The device executes the transfer without further attention. The LH-DH/11 writes received data directly into the user's input buffer and reads data to be transferred over the link directly from the user's output buffer. When a data transfer is complete, the device signals the LHDHD. The LHDHD then replies to the user's request.

The stty system call may be used by the user process for special functions. At present, the only special function implemented is flushing of all outstanding write requests.

2.4.4 Video Information Projection Module. The Video Information Projection Module (VIPM) provides UNIX processes with an interface to WWMCCS standard Honeywell VIP terminals. It performs protocol and data formatting functions unique to VIP terminals. The VIPM supports 786W VIPs and 7705W VIPs. The Print Page Adapter (PPA) for VIPs is also supported.

The VIPM interfaces with the Terminal-to-Host Protocol Service Access Module (THPSAM) and the DV11 Device Driver (DV11D). The VIPM allows a user process to treat UNIX TTY terminals and VIPs in the same manner. In the current WNFE configuration, THPSAM is the only user process. VIPs connected to the WNFE use the THPSAM to access both the local host and the network.

In addition to transmitting data between a user process and VIP terminals (via the DV11D), the VIPM notifies the user process of the following events:

a. VIP operator login

b. VIP operator sending BREAK

c. VIP operator disconnect

d. Use of the PRT key for transmission

The VIPM interprets the following GRTS-II (General Remote Terminal Supervisor) commands, which can be in either upper or lower case, when they are transmitted from the terminal:

a. $*$LOG24

b. $*$LOG22L

c. $*$5

d. $*$6

e. $*$B

f. $*$BRK

g. $*$D

h. $*$DIS

These commands have the same meaning to the VIPM as they do to the WWMCCS Datanet communications processor.

The 786W VIP can only display upper case characters. The VIPM maps all lower case characters being sent to a 786W VIP into upper case. The 7705W VIP can display both upper and lower case characters, so no mapping is required. The VIPM also responds to an indication from the user process that a data buffer contains special information. This is used to interpret special headers for data routed to the Page Printer Adapter (PPA). The PPA provides a method of obtaining a hard copy of the data displayed on the VIP screen or data sent directly from the user process.

The 786W and 7705W VIPs can operate in either the contention or poll/select mode. In the contention mode, each VIP terminal is connected to the VIPM by a dedicated hardware line. In the poll/select mode, up to 32 terminals can be addressed on each hardware line by using Multiple Interface Units (see Figure 2-10). However, communication hardware and response time constraints limit the practical maximum to 4 - 6 terminals in simultaneous operation.

VIP terminals are operated in the half-duplex mode. VIPs configured for poll/select mode are polled to see if they have any data to be sent to the user process. If not, the terminal sends a quiescent frame (Q-frame). When the VIPM sends data to a polled VIP, it prepends a select header, which determines which VIP terminal will recognize the data.

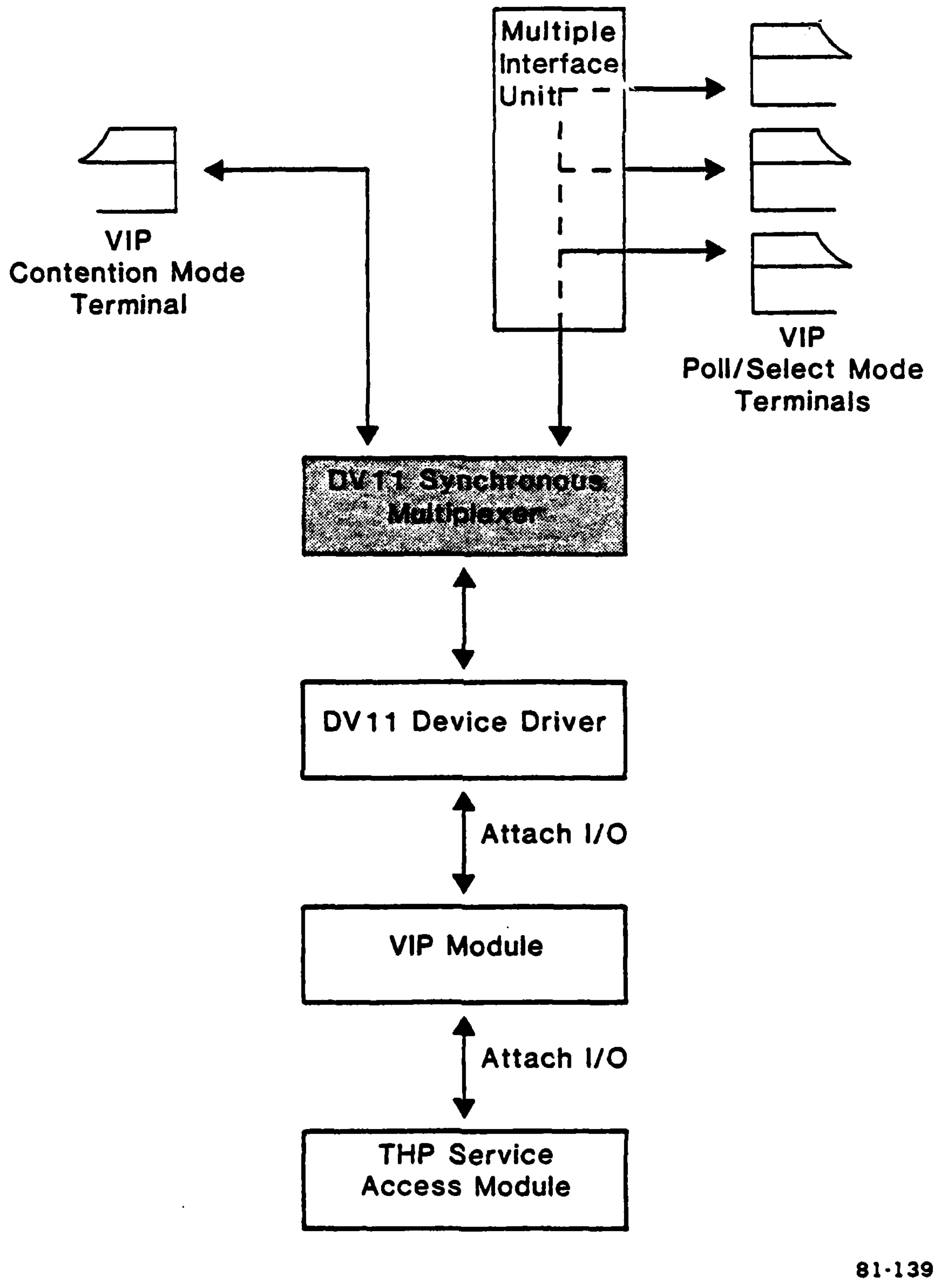

Multiple Interface Unit
VIP Contention Mode Terminal
VIP Poll/Select Mode Terminals
DV11 Synchronous Multiplexer
DV11 Device Driver
Attach I/O
VIP Module
Attach I/O
THP Service Access Module
81-139

Figure 2-10. VIP Module Communications

Contention mode VIPs are connected singly on half-duplex lines. Contention mode VIPs and the VIPM continually exchange Q-frames during idle periods to turn the line around. When either has data to transmit, data is sent instead of a Q-frame.

2.5 Host-to-Front End Protocol Modules

This subsection briefly describes the modules that implement the channel and service access levels of the Host-to-Front End Protocol. The link level is implemented in the LH-DH/11 Device Driver (*LHDHD*) described in subsection 2.4.3. The channel level is implemented in the Channel Protocol Module. The service access level is implemented in two modules, one for each service provided.

2.5.1 Channel Protocol Module. The Channel Protocol Module (CPM) provides a communication medium between service access modules in the host and service access modules in the WNFE. This communications medium appears to the host and WNFE service access modules as a set of host-to-front end virtual channels. Data and controls are transmitted over a virtual channel in the form of HFP messages.

The CPM provides for duplicate-free and loss-free message exchange over these virtual channels. The data may be either ordered with flow control or unordered without flow control. The CPM communicates with service access protocol modules and the *LHDH* Device Driver (Link Level HFP) via Attach I/O as shown in Figure 2-11.

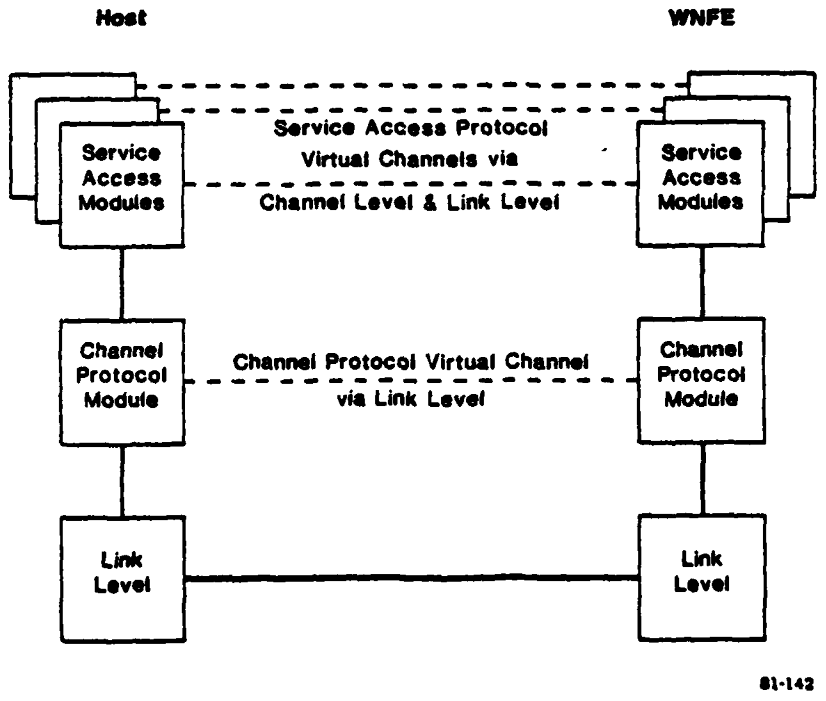


Figure 2-11. Host-to-Front End Protocol Architecture

There is one Attach I/O stream for each virtual channel connecting a WNFE service access module and the CPM. A service access module may establish multiple streams with the CPM. Also the CPM supports streams from several service modules simultaneously.

There is only one Attach I/O stream between the CPM and the LHDHD. It is established when the CPM and LHDHD are initialized.

The CPM performs the following functions:

a. Establishes virtual channels

b. Monitors and changes the condition (state) of the channel

c. Receives messages from the WNFE service access modules, multiplexes the messages, and sends them to the host CPM via the HFP link level

d. Receives messages from the host via the link level, demultiplexes the messages, and routes them to the proper WNFE service access modules

e. Detects and allows for out-of-order messages

f. Regulates the rate at which HFP messages are transmitted over the virtual channel

g. Terminates virtual channels

2.5.1.1 **HFP Messages.** Each HFP message has two parts: a header and text. The header contains the information necessary for executing the channel protocol. The text contains a service module message. The text part may be empty in certain cases. A type code contained in the HFP header determines the message type. The types are

a. Begin - used for the creation of a virtual channel

b. Transmit - used for the transmission of flow controlled data

c. Execute - used for the transmission of non-flow controlled data

d. End - used for the termination of virtual channels

e. Nop - used to clear the link communication channel.

The type code also specifies one of two variants:

a. The Command variant is used by a module to command the module receiving the message to perform a function.

b. The Response variant is used by the module commanded to perform a function to report success or failure in carrying out the command.

2.5.1.2 **CPM Functional Components.** The CPM consists of a channel interface, channel machines, and a multiplexer/demultiplexer (see Figure 2-12).

a. The channel interface provides the service access modules with access to the channel machines.

b. Each channel machine maintains the state of one end of a virtual channel. A channel machine consists of a channel interface, a state machine, and a transmission controller.

   1. The channel interface receives messages from a service access module or a state machine, performs consistency checks on them, and passes them on to the state machine or service access module respectively.

   2. The state machine maintains the state of its end of the channel. The state machine changes state in response to the messages received by the channel machine.

   3. The transmission controller provides message flow control, duplicate message detection, and out of order message detection.

c. The multiplexer/demultiplexer interleaves messages coming from the channel machines and formats them for transmission via the link level. It filters messages coming from the link level and routes them to their respective channel machines. Filtering consists of checking the messages for consistency. Messages failing these checks are filtered out of the data stream. Because the filtered out messages are not acknowledged, they will be retransmitted by the sender.

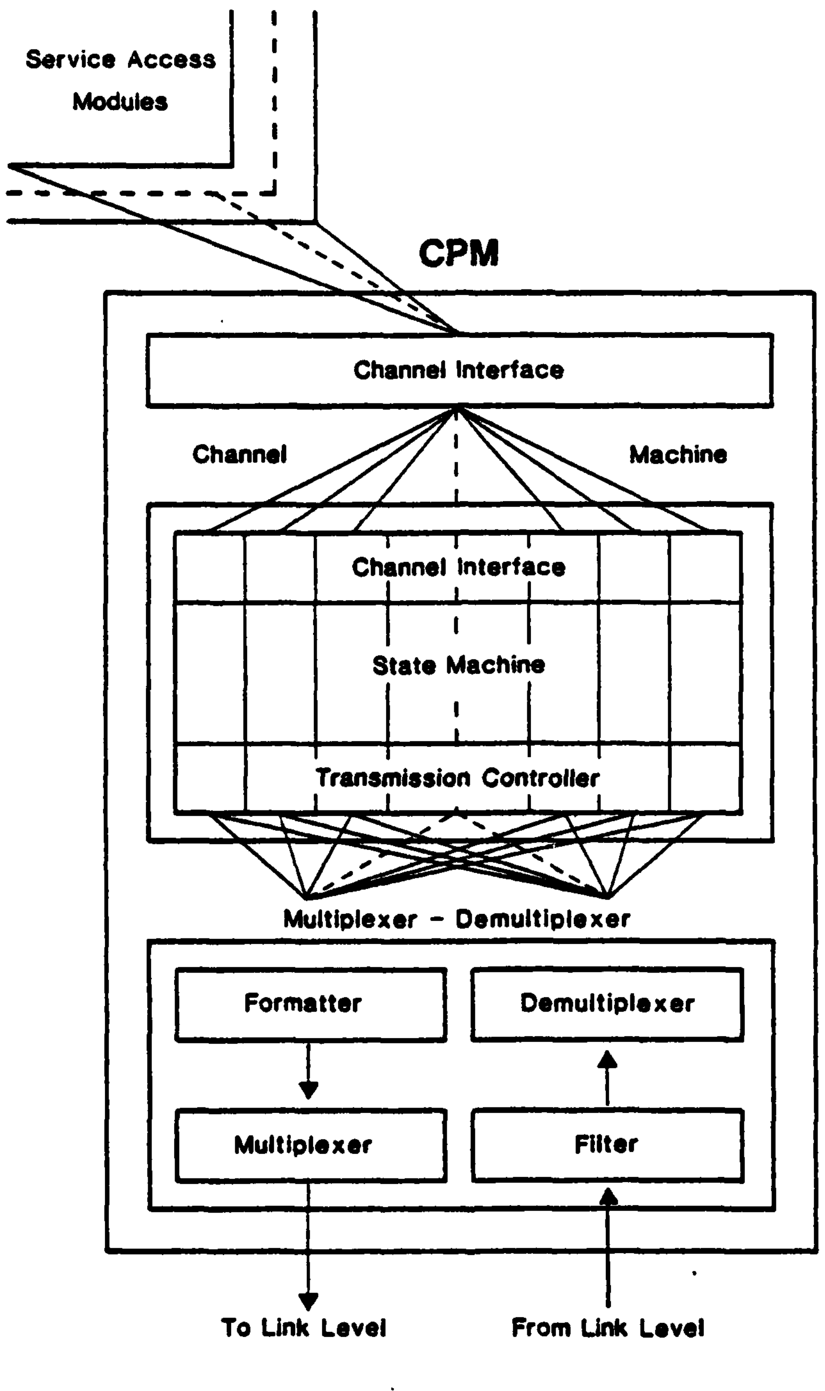


Figure 2-12. Channel Protocol Functional Diagram

2.5.1.3 Virtual Channels. A virtual channel can be initiated from either the host or WNFE. A service access module initiates a channel by requesting its CPM to send a Begin Command to the other CPM.

Each HFP message is assigned a sequence number. These sequence numbers are used for acknowledging receipt of messages, ensuring in-order reception, and controlling message flow. The CPM provides both flow controlled and uncontrolled transmission between it and the host CPM.

Message flow is controlled on the CPM-to-CPM virtual channel and at the CPM-to-service access module interfaces. In both cases, flow control is achieved by the receiver informing the sender of the number of messages the receiver is prepared to receive. The sender regulates transmissions accordingly.

An uncontrolled data stream provides a means to expedite data transfer with respect to the controlled data stream. In addition, an attention flag to interrupt the service access module can be associated with expedited messages.

Either the host or WNFE service access module can initiate termination of a channel. Channels may be terminated in two ways: flushing or non-flushing. A flushing termination causes the channel machine to discard all queued data and enter a terminating state. A non-flushing termination causes the channel machine to send all queued data before entering a terminating state.

2.5.2 THP Service Access Module. The Terminal-to-Host Protocol Service Access Module (THPSAM) implements the THP service access level of HFP and portions of the Terminal-to-Host Protocol (THP). The primary purpose of the Terminal-to-Host Protocol is to provide network services to terminal users. The protocol allows network communication between two terminals, between a terminal and a terminal-oriented process, and between two terminal-oriented processes.

Functions performed by the THPSAM to implement the Terminal-to-Host Protocol include

a. Making terminals and processes appear similar

b. Translating control characters such as IP (interrupt process), AO (abort output), EC (erase character), and EL (erase line) from the local character set to the network standard character set and vice versa

c. Providing commands to set terminal control parameters

d. Providing connection management controls to include opening, closing, interrupting, flushing data, and entering and exiting the send-receive mode

On the local terminal user side, the THPSAM communicates with the Channel Protocol Module (CPM), the UNIX terminal handler, and the Video Information Projection Module (VIPM). On the network side, it communicates with the Terminal-to-Host Protocol Module (THPM). All communications between THPSAM and its neighbors is via Attach I/O (see Figure 2-13).

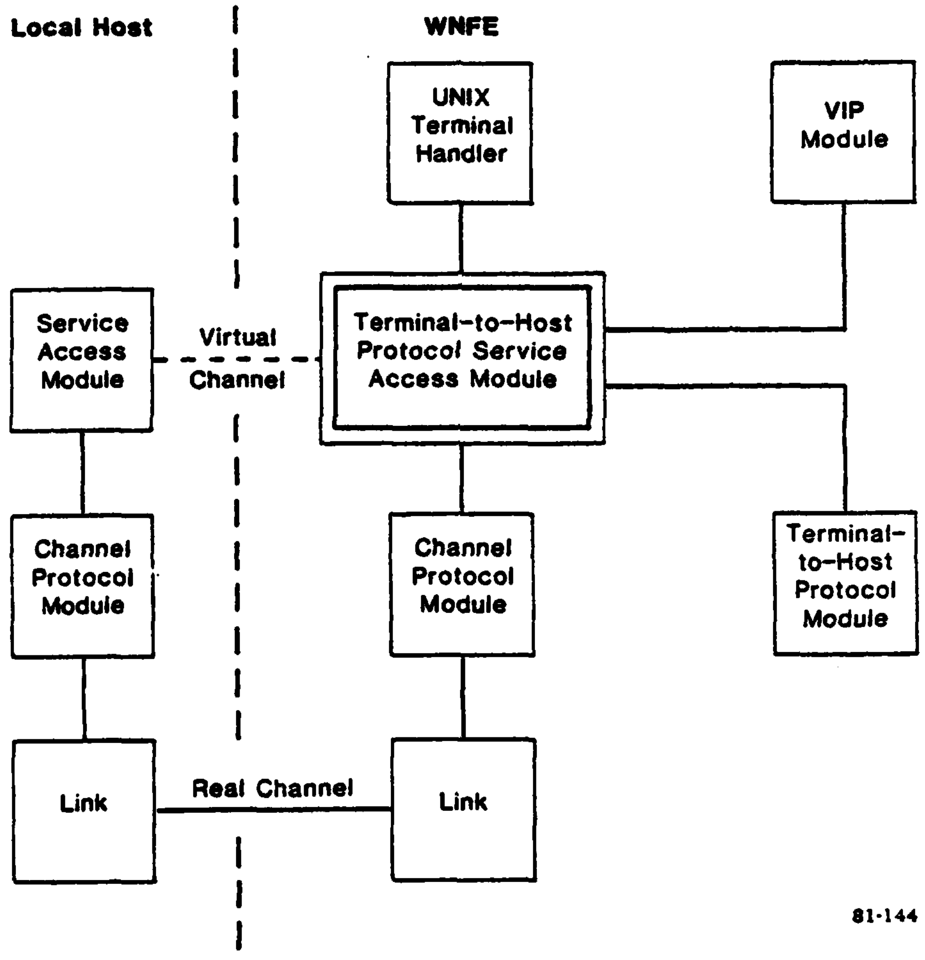


Figure 2-13. THPSAM Communications

The THPSAM translates

a. HFP messages into suitable input messages to the THPM and THPM output messages into HFP messages

b. HFP messages into UNIX terminal handler data representations and vice versa

c. UNIX terminal handler data representations into suitable input messages to the THPM and THPM output messages into UNIX terminal handler data representations

2.5.2.1 **THPSAM to CPM Communication.** THPSAM uses the Channel Level HFP, implemented by the CPM, to communicate with the host terminals and host processes that serve terminal users. THPSAM establishes one HFP virtual channel for every WNFE communication channel involving a local host terminal or a local host application program. Initiating an HFP virtual channel can occur two ways:

a. If a remote user or local WNFE user requests a connection to a local host application program, the THPSAM requests the CPM to open an HFP channel.

b. If a local host terminal user requests a virtual channel to the THPSAM, the Virtual Terminal Program on the host requests the local host's Host-to-Front End Protocol Module to open an HFP channel.

Once a virtual channel is established, the THPSAM can request the CPM to transfer text to and from the local host via HFP messages. It can also send and receive out-of-band control messages via HFP messages.

Termination of an HFP virtual channel can be initiated by the application program in the local host or by the THPSAM in the WNFE.

2.5.2.2 THPSAM to Terminal Handler Communication. The UNIX terminal handler implements the terminal-specific protocols for Teletype (TTY) type terminals connected to the WNFE. Features specific to the UNIX terminal interface include

a. The terminal handler informs the THPSAM when the user presses the **BREAK** key on the terminal.

b. THPSAM can direct the terminal handler to

1. Change the data transmission rate

2. Turn echoing on or off (if this is not a <u>local copy</u> terminal)

3. Change the Line-delete character and the Character-delete character

4. Obtain the current parameters of a terminal

One Attach I/O stream is created for each terminal logon session. The THPSAM initiates the stream, sets terminal-specific parameters for the terminal, and waits for a user to type a carriage-return (CR), signifying a desire to log in to WNFE.

When a CR is typed, THPSAM initiates its logon procedure to obtain the user-specific parameters. Until the user has been fully validated, the THPSAM sends all terminal input to the Authentication and Accounting Module (AAM), and forwards messages from the AAM to the terminal. In this way, the AAM converses with the user until he has been fully logged in. At this point, the AAM informs the THPSAM that the user is validated. The user can then enter commands to open a connection to a host.

The THPSAM closes the terminal Attach I/O stream if

a. The user types the **bye** command.

b. The user's inactivity timer expires.

2.5.2.3 THPSAM to VIP Module Communication. The THPSAM communicates with and processes data to and from VIPs in essentially the same manner as described above for TTY terminals. There are, however, a few differences:

a. The THPSAM cannot set VIP terminal parameters.

b. The THPSAM determines the type of VIP by looking at the *$LOG message typed by the user during logon.

c. Requests and replies carrying data have a flag bit to indicate the end of a line or screen of data to or from the user.

d. The THPSAM can inform the VIPM that data includes a special header to be decoded for VIP Page Printer Adapter (PPA) operations.

Data received from the VIPM appears exactly as TTY terminal data.

2.5.2.4 THPSAM to THP Module Communication. THPSAM maps HFP messages containing THP requests into requests to the THP Module. The THP Module implements the Terminal-to-Host Protocol using Transmission Control Protocol (TCP) connections as its communication medium to the network.

THPSAM creates one Attach I/O stream to the THPM for each THP connection. The resulting request delivered to the THPM contains the information required to establish the underlying TCP connection. There are two scenarios for establishing a THP connection:

a. A local host user or a WNFE user can request network connection. Once the AAM has validated the network request, the THPSAM requests a THP connection, specifying a TCP INIT (open the connection immediately). When the connection opens, the THPSAM carries on a brief conversation over the TCP connection with the remote THPSAM. Information exchanged at this time includes

   1. The type of the user's terminal

   2. The default inactivity timeout period

   Once this initial negotiation is done, the user can send and receive data over the connection.

b. The AAM can tell the THPSAM to listen for a remote user to request access to the WNFE. In this case, the THPSAM specifies a TCP LISTEN (wait until a remote user does an INIT) in the request to the THPM for a TCP connection.

When the connection initiated by the remote user opens, the THPSAM first carries on the initial negotiation conversation described above. It then queries the AAM to verify that the remote user is authorized to make the connection. If the AAM responds positively, the THP connection is considered fully open. At this point, data can be sent to and from the remote user.

The THPSAM requests closing of the network connection in response to

a. A bye or a close command typed by the local user (either WNFE or host)

b. An HFP End Command from the local host

c. An indication from the THPM that the remote THPSAM is closing the connection

In any of these situations, the THPSAM requests the THPM to close the connection. The THPM then closes the underlying TCP connection.

2.5.3 TCP Service Access Module. The Transmission Control Protocol Service Access Module (TCPSAM) provides host processes with access to the Transmission Control Protocol Module (TCPM), and hence to the network. The TCPSAM communicates with service access modules in the host via virtual channels provided by the channel level and link level modules (see Figure 2-14).

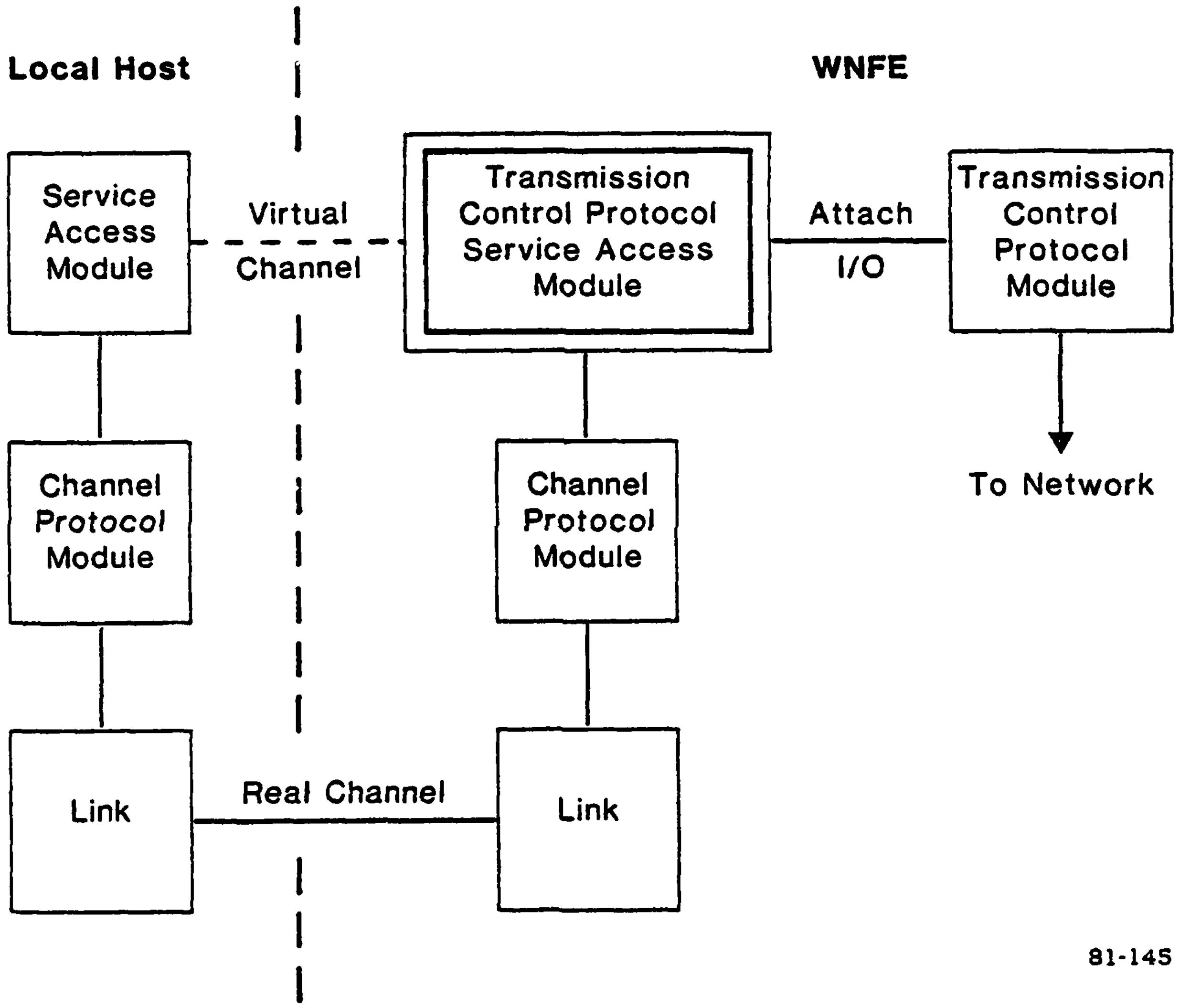


Figure 2-14. TCPSAM Communications

The TCPSAM provides for the transfer of data and out-of-band messages (such as TCP connection status requests and Urgent indications) between the Channel Protocol Module (CPM) and TCPM.

For each HFP message received from the host, the TCPSAM determines the command type and extracts any data from the text field of the message. Based on the HFP command type, the TCPSAM may send data to the TCPM, open or close Attach I/O streams, or perform special requests. Special requests are used for functions other than data transfer, such as obtaining TCP connection status and changing connection security levels.

2.5.3.1 **HFP Channel Establishment.** The TCPSAM communicates with its neighboring modules via Attach I/O. When first initialized, the TCPSAM attempts to open Attach I/O streams to the CPM. The desire on the part of a host process to open a TCP connection is received as an Open reply from the CPM. The reply includes the information the needed to open a TCP connection.

The TCPSAM builds a open command string from the received reply and passes it to the TCPM in an Attach I/O Open request.

The TCPM then uses the command string in this request to establish a TCP connection. The TCPM replies to the TCPSAM Open request when the TCP connection is established. The TCPSAM then requests the TCP connection status, and The TCPM replies with the TCP connection status.

At this time the the TCPSAM sends a request containing the connection status to the CPM. With the TCP connection is established, a virtual communications path (HFP channel) exists between the host process and the TCPSAM, allowing the host process to access the TCP connection.

The TCPSAM receives data from the host process via HFP messages. The data is extracted from the message and passed to the TCPM. When data is received from the TCPM, the TCPSAM incorporates it into an HFP message and sends it to the CPM. TCPSAM does not perform any transformations on the data flowing in either direction.

The NFE is a 16-bit machine while the host is a 36-bit machine. Therefore, the TCPSAM data buffer size is a common multiple of 16 and 36, which facilitates use of the Service Access Protocol implementation in the host and prevents message splitting from occurring within a host word.

2.5.3.2 Out-of-Band Message Processing. An out-of-band message is sent when an Urgent indication is to be sent to the TCPM or when the host process wants to obtain or change the status of a TCP connection.

If a message containing an Urgent indication is received from the CPM, the TCPSAM sends a special request to the TCPM containing the Urgent indication. All subsequent requests sent to the TCPM have an Urgent indication until a message containing an End-of-urgent indication is received from the CPM. The data in this message will be sent to the TCPM with an End-of-urgent indication, notifying the TCPM that there is no more urgent data.

If an Urgent indication is received from the TCPM, the TCPSAM sends an out-of-band message indicating Urgent to the CPM. All subsequent messages sent to the CPM have an Urgent indication until the TCPM indicates End-of-urgent. The End-of-urgent indication is passed on the the CPM.

2.5.3.3 HFP Channel Closing. The channel will be closed if

a. The host service access module sends an HFP End Command.

b. The remote TCPM closes the TCP connection.

The host process can request a flushing or non-flushing close. A flushing close causes immediate termination of the connection and any data in transit to the TCPM is discarded. For a non-flushing close all data in transit is sent before the TCP connection is terminated.

2.6 Network Protocol Modules

There are five network protocol modules.

1. The Terminal-to-Host Protocol Module (THPM) allows a variety of terminals and terminal-oriented processes to communicate over the network.

2. The Transmission Control Protocol (TCPM) provides a connection-oriented, reliable, end-to-end communication facility to remote WNFEs and remote hosts.

3. The Internet Protocol - 1822 Protocol Module (IP1822M) provides a datagram service across multiple networks and implements a portion of the BBN 1822 protocol.

4. The LH-DH/11 Device Driver (LHDHD) together with the device provide a reliable data link to the network Interface Message Processor (IMP) by implementing most of the BBN 1822 protocol.

The relationships of these module to each other and to corresponding modules in a remote WNFE are shown in Figure 2-3 on page 2-8.

2.6.1 Terminal-to-Host Protocol Module. The Terminal-to-Host Protocol Module (THPM) implements THP communication channels using TCP connections as its network communication medium. A THP communication channel connects local and remote terminals and/or terminal-oriented application programs. From the point view of the THPM, each THP communication channel consists of an Attach I/O stream from the THP Service Access Module (THPSAM) to the THPM and an Attach I/O stream from the THPM to the Transmission Control Protocol Module (TCPM) as shown in Figure 2-15.

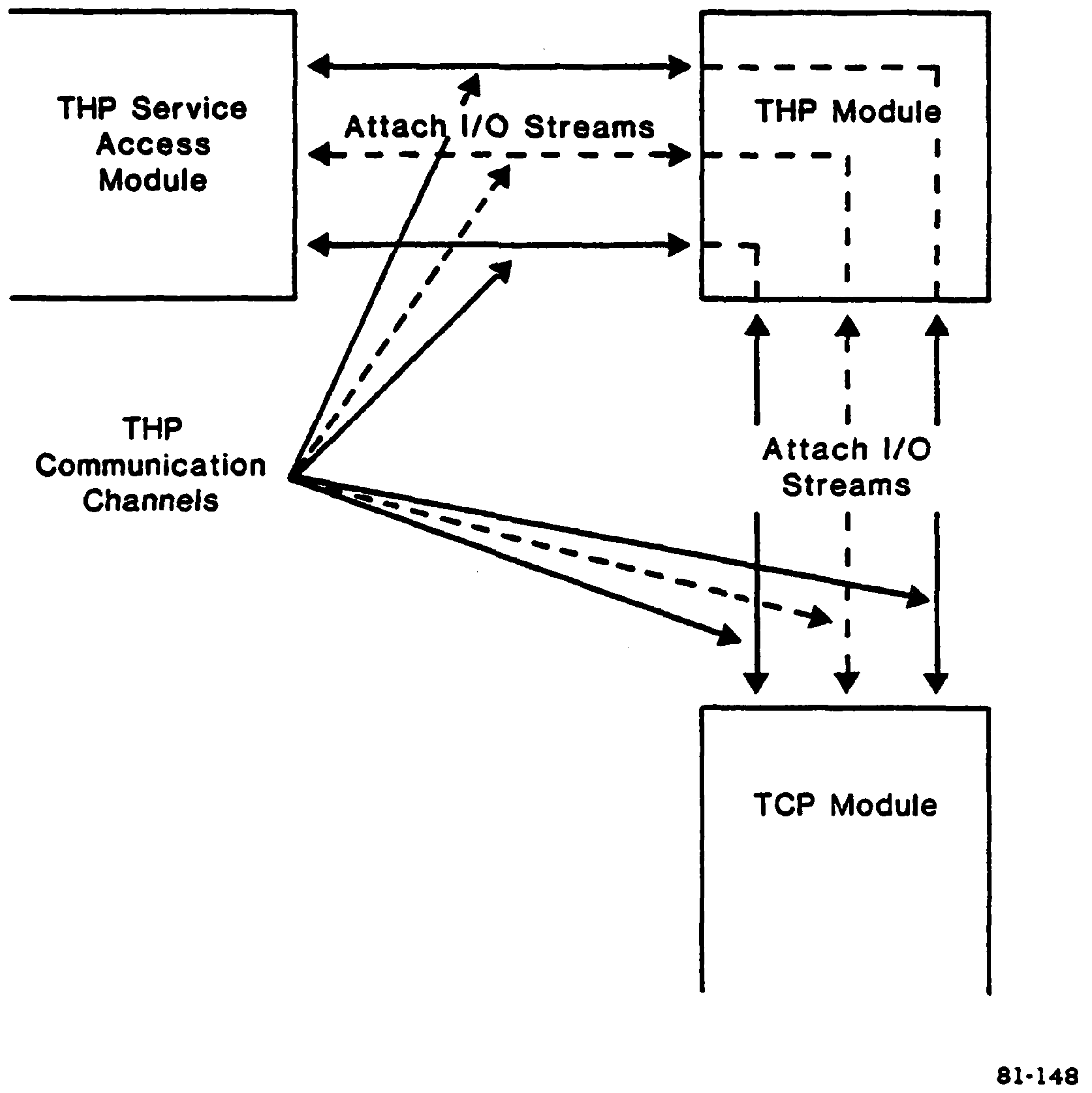


Figure 2-15. THP Communication Channels

2.6.1.1 THP Module to THPSAM Communication. For each THP communication channel to be established, The THPM receives a request from the THPSAM to open an Attach I/O stream and a TCP connection. The request contains the information required to establish the TCP connection. When the connection attempt is completed, the THPM replies to the THPSAM indicating whether the TCP connection attempt was successful. If successful, the Attach I/O stream and the TCP connection is open.

Once the Attach I/O stream is open, the THPM can respond to requests from the THPSAM to transfer text and option negotiations between the THPM and the THPSAM. For each request, the THPM replies to the THPSAM when the transfer is complete.

The THPM can also respond to special I/O requests from the THPSAM. These special requests may

a. Request the status of the TCP connection

b. Set the security parameters of the TCP connection

c. Send THP controls

d. Request receipt of THP controls

For each request, the THPM replies to the THPSAM after the action has been performed.

The THPSAM can also request the THPM to close the Attach I/O stream and the TCP connection. When the connection has been closed, the THPM replies to the THPSAM.

2.6.1.2 THP Module to TCP Module Communication. For each request received to open a THP channel, the THPM attempts to open an Attach I/O stream to the TCPM. The Open request contains the information required to establish a corresponding TCP connection. When the connection attempt is completed, the TCPM replies indicating whether the attempt was successful. If the attempt was successful, the Attach I/O stream is open and the TCP connection is established.

Once the stream and the TCP connection is open, the THPM can request the TCPM to transfer data to and from the TCPM and the remote THPM. For each request, the TCPM replies when the transfer has been performed.

In addition to requesting data transfer, the THPM can

a. Request the status of the TCP connection

b. Set the security parameters of the TCP connection

c. Inform the TCP module that all previously sent data is urgent

d. Request receipt of a TCP beginning-of-urgent indication

For each request, the TCPM replies to THPM after the action has been performed.

The THPM-to-TCPM Attach I/O stream and the TCP connection is closed at the request of THPM. The TCPM replies to the THPM when the connection has been closed. If the TCP connection is closed by the remote THPM, the TCPM informs the local THPM that it should close the Attach I/O stream.

2.6.2 Transmission Control Protocol Module. The WNFE Transmission Control Protocol Module (TCPM) implements the DoD Standard Transmission Control Protocol (TCP).

TCP provides a connection-oriented, highly reliable host-process to host-process communication service for packet-switching networks. TCP provides facilities for reliable data transfer, connection management, flow control, multiplexing, and precedence and security functions.

2.6.2.1 **Basic Data Transfer.** The TCPM transfers data for its users by packaging some number of octets into segments for transmission through the internet system. Each transmitted segment has a header containing such information as the connection identifier, sequence numbers, and controls.

The end-of-letter (EOL) flag is available as a mechanism for indicating the end of a logical group of data. The EOL causes the TCPMs at both ends of the TCP connection to promptly forward and deliver data up to that point to the receiver.

2.6.2.2 **Opening and Closing Connections.** A user level process initiates the establishment of a connection with an Open request. In the current WNFE configuration, the user level processes are the Terminal-to-Host Protocol Module (THPM), the Transmission Control Protocol Service Access Module (TCPSAM), and the Authentication and Accounting Module (AAM). The request contains all the information needed by the TCPM to establish the connection.

Two types of Open requests may be used: a passive listening type or an active type. The listening type is used for processes that accept connections from remote users. In this case the TCPM sends nothing to the network. A connection will not be made until a request for connection to a process comes from a remote user via the network. The active type of open request causes the TCPM to actively attempt a connection to the destination address.

Each TCP connection is uniquely identified by a connection identifier. The connection identifier is divided into six address fields containing the following information:

a. Local network address

b. Local TCP address

c. Local port address

d. Remote network address

e. Remote TCP address

f. Remote port address

Ten states describe the condition of a connection. The TCPM makes a majority of its decisions concerning communication with the network on the basis of the current connection state. Several state transitions occur in opening a connection as the protocol requires an exchange of synchronizing segments before the connection is established. When the connection is established or when the connection attempt fails, the TCPM replies to the user's Open request.

A user level process initiates the closing of a connection. All data that was to be transmitted is sent and acknowledged before the transmitting TCPM closes the connection unless a flushing closes is requested. For a flushing close, data not yet transmitted is discarded. Again, several state transitions occur during closing to ensure all data in transit has been handled properly.

It is possible for two TCPMs to disagree on the state of a connection. For example, one TCPM may think that a connection is established while the other thinks that it is closed. Although this is an unlikely event, recovery is provided for. The TCPM that is first to detect this condition resets the connection and sends a segment containing a reset indication to the other TCP.

2.6.2.3 **Reliability.** Once a connection is established, data can be transmitted and received by the TCPM. Data is transmitted in the form of TCP segments. The network does not guarantee error-free transmission of segments. Four transmission errors may occur:

a. Segments may arrive out of order.

b. Segments may be missing.

c. Segments may be duplicated.

d. Segments may contain bit errors.

The TCPM detects these errors. Retransmission by the sending TCPM ensures data reliability.

2.6.2.3.1 **Sequence Numbers.** Sequence numbers are used by TCP to detect occurrence of the first three errors listed above. Sequence numbers are passed between TCPMs in segment headers. The first segment transmitted by a TCPM when establishing a connection is always the synchronize (SYN) segment. The SYN segment contains an initial sequence number. All data and control bits subsequently transmitted over the connection are assigned sequence numbers based on the value of the initial sequence number.

2.6.2.3.2 **Acknowledgements and Timeouts.** The TCPM stores all transmitted data on a retransmission queue until reception is acknowledged. A receiving TCPM acknowledges the receipt of ordered data by sending an acknowledgement sequence number to the transmitting TCPM. If the transmitting TCPM does not receive an acknowledgement within a few seconds of transmitting data, the data is retransmitted. When the transmitting TCP receives an acknowledgement sequence number the acknowledged data is removed from the retransmission queue and discarded.

2.6.2.3.3 **Out of Order Processing.** All data received by the TCPM must be given to the user level process in order. The TCPM only acknowledges data that has been received in order. The TCPM takes full responsibility for ordering all the data it receives from a remote TCPM.

2.6.2.3.4 **Checksums.** The contents of a segment can be corrupted by the network. To detect this error, the transmitting TCPM computes a checksum for each transmitted segment and puts it in the segment header. The receiving TCPM also computes the checksum and compares its computed checksum with the checksum in the received segment header. If there is a discrepancy the received segment is discarded. Not acknowledging the corrupted segment ensures retransmission by the sending TCPM.

2.6.2.4 **Flow Control.** Flow control ensures that a sending TCPM will not overrun a receiving TCPM's buffer space. A receiving TCPM informs a transmitting TCPM of how much data it is able to receive by transmitting the number of data bytes it is able to accept. The transmitting TCPM transforms the number of data bytes into a range of sequence numbers called a window. The transmitting TCPM restricts data transmissions to stay within this window.

2.6.2.5 **Addresses, Precedence and Security.** A completely specified connection identifier describes the address of both communicating processes. The combination of the network and TCP address fields identifies a particular host. The port address field identifies a process at the host. Port addresses enables hosts and processes to have multiple connections. One user level process can communicate with several other remote user level processes simultaneously. Also a local process and remote process can communicate over more than one connection simultaneously.

A user level process can specify all address fields when requesting TCP connection or specify only some of the fields. Any unspecified port address will be assigned by the Authentication and Accounting Module (AAM) when the TCPM requests connection authentication. The allowable port addresses, security, precedence, and transmission control code (TCC) information concerning the user level process are authenticated by the AAM. Once a connection is established, the security, precedence, and TCC cannot be changed.

2.6.2.6 **Urgents.** With the urgent mechanism a transmitting user level process informs a receiving user level process that there is urgent data. It is assumed that upon receipt of an urgent, the receiving user level process will enter a "read fast" mode to expedite handling of the urgent data. Urgent provides a type of out-of-band communication mechanism.

2.6.3 **Internet Protocol - 1822 Protocol Module.** The Internet Protocol - 1822 Protocol Module (IP1822M) implements the DoD Standard Internet Protocol and a portion of the Bolt, Beranek and Newman 1822 Protocol.

The Internet Protocol is designed for use in interconnected systems of packet-switched computer communication networks. The Internet Protocol provides facilities for transmitting blocks of data, called datagrams, from sources to destinations in possibly different networks. Each datagram is treated as an independent entity. There is no concept of an end-to-end virtual channel or connection at this protocol level.

The Internet Protocol provides for fragmentation and reassembly of long datagrams for transmission through networks with a small maximum packet size. Fragmentation will normally be performed only by gateways to other networks. Since the WNFE does not act as a gateway, it does not support fragmentation, but does support reassembly of fragmented datagrams.

Dual homing is also supported. Dual homing permits a WNFE to communicate over the network using more than one network access line to the network. In other words, the WNFE can be linked to more than one IMP. This feature increases network communication reliability and allows increased bandwidth through the use of routing algorithms to balance the load on the access lines.

As shown in Figure 2-3 on page 2-8, the IP1822M interfaces with the TCPM on the user side. On the network side, the IP1822M interfaces with the LH/DH Device Driver. Communication over both of these interfaces is via Attach I/O.

In response to an Open request from the TCPM, the IP1822M opens all available network access lines. In response to a Close request, the IP1822M flushes all outstanding requests but does not actually close the access lines.

Once the access lines are open, the TCPM can request the transfer of TCP segments over the network. A request to send a TCP segment includes the information required by the IP1822M to construct the IP header. The IP1822M packages the TCP segment as a datagram and sends it to the the network using the 1822 protocol. The IP1822M replies to the requests after receiving an acknowledgement of reception of the datagram from the local IMP.

The TCPM also issues requests to receive TCP segments from the network. The IP1822M responds to these requests by passing TCP segments received from the network to the TCPM. The IP1822M includes in the reply a summary of the information contained in the received headers.

That portion of the 1822 protocol relating to constructing and interpreting Host-to-IMP headers is implemented in the IP1822M. The portion of the 1822 protocol relating to the actual data transfer over the access lines to the Interface Message Processor (IMP) is implemented in the LHDHD and in the device itself.

2.6.3.1 **Dual Homing.** The IP1822M performs a multiplexing/demultiplexing function as part of dual homing. There may be more than one network access line, and all the access lines may be simultaneously active. Data flow for a single TCP connection may use any or all access lines.

Each network access line is assigned a unique id. That is, the access lines are treated as distinct WNFEs. Therefore, dual-homed WNFEs have more than one id, one for each access line. All such WNFEs have one of their ids declared to be their <u>well-known</u> id. Other ids are called <u>alias</u> ids. Only the well-known id is used in communicating over the TCPM-to-IP1822M interface.

The IP1822M references a dual homing table of id numbers (addresses) when handling incoming and outgoing datagrams. For incoming datagrams, the IP1822M maps the source id contained in the internet header to the well-known id by referring to this table.

Three ids (addresses) are involved when sending a datagram to the network: (1) the id that determines which local line to send the datagram on, (2) the id that determines which local line the remote WNFE should use when sending datagrams to the local WNFE, and (3) the id of the remote line to address. The IP1822M may change both the source and destination addresses supplied by the user for outgoing datagrams. The details of dual homing are invisible to the TCPM and all other higher level users. They are not concerned with alias ids or access lines.

2.6.3.2 **Fragment Reassembly.** If a datagram from a remote WNFE on another network passed through a network with a small maximum packet size, the datagram will have been fragmented. Consequently, the WNFE will receive two or more datagrams that are only datagram fragments. The IP1822M uses the information in the datagram headers to reassemble such fragmented datagrams. Only after a datagram is complete does the IP1822M extract the TCP segment and pass it to the TCPM.

2.7 WNFE Control and Monitoring Modules

Two WNFE modules provide for the control and monitoring of the WNFE system while it is in operation.

a. The Authentication and Accounting Module (AAM) consolidates all access to configuration files, authentication files, and log files in one module. It starts all the other modules when the system is started and restarted. It also continuously monitors the other modules, resources, and status of WNFE terminals.

b. The Operator and Site Administrator Interface Module (OSAIM) allows authorized operators, such as the system Operator, Site Administrator, and Site Security Officer, to monitor and control activity on the system.

2.7.1 Authentication and Accounting Module. The AAM starts and restarts all the other WNFE modules and is the only module with direct access to the WNFE file system. All other WNFE modules that must store or retrieve file data must do so via requests to the AAM. Figure 2-16 shows the relationship of the AAM to some of the other WNFE modules and to the WNFE file system.

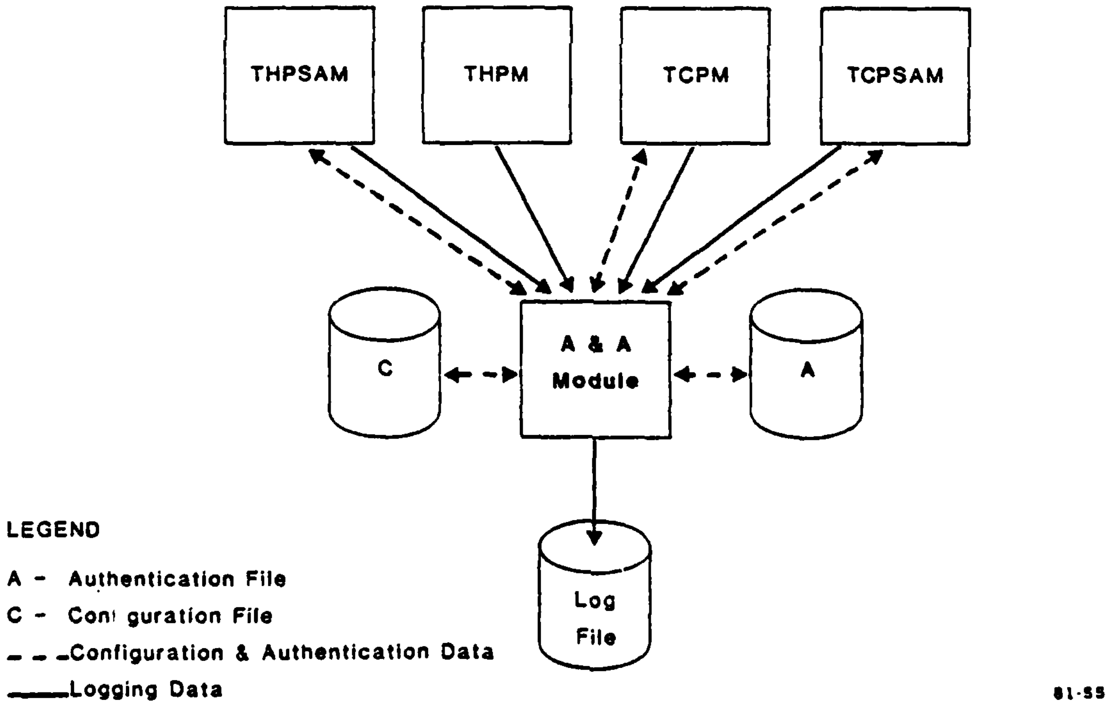


Figure 2-16. Relation of AAM to Modules and File System

This section describes the interactions between the AAM and

a. The WNFE protocol and service modules

b. The Operator/Site Administrator Module (OSAIM)

c. The host, to include

   1. Transfer of authentication information from the host to the AAM

   2. Transfer of audit trail information from the WNFE Log File to the host

d. An AAM in a remote WNFE

The AAM performs four authentication and accounting functions:

a. It controls access by users and application programs to system and network resources (authentication).

b. It produces an audit trail of the access and use of these resources (accounting).

c. It maintains authentication information.

d. It maintains and monitors hardware and software configuration.

The AAM also supports the following WNFE functions:

Single logon.

Single logon enables a user to gain access to the entire network by logging on only at the local host.

Job spawning.

Job spawning allows a job on a network host to be started on demand.

Dynamic reconfiguration.

This function enables authorized operators to modify device configuration parameters without rebooting the WNFE.

Log transfer.

This function transfers the WNFE audit trail data to the host.

Operator interface.

This function enables the operator and site administrator to control and monitor the WNFE.

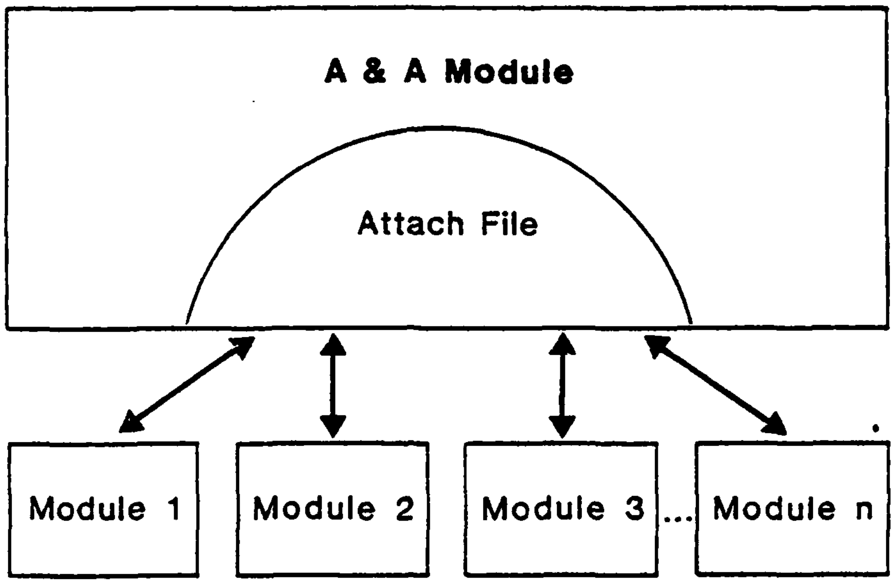
Figure 2-17 shows the authentication and accounting requirements for each of the WNFE modules and the UNIX operating system.

Module	Accounting		Authentication	Configuration
	Connection Status (Audit)	Module Status		
OSAIM	X	X	X	
TCPSAM	X	X	X	
THPSAM	X	X	X	X
CPM		X		
THPM	X	X		
TCPM	X	X		X
IP1822M		X		X
VIPM		X		X
UNIX		X		X

Figure 2-17. Authentication and Accounting Requirements

2.7.1.1 Starting WNFE Modules. When the UNIX operating system is loaded, it starts an initialization routine called init. The AAM is the only WNFE module started by init. The AAM starts all other WNFE modules, and has complete responsibility for monitoring the execution of the module. It restarts any module that crashes. If the AAM crashes, init reboots the WNFE.

2.7.1.2 AAM Data Paths. The AAM creates an Attach I/O file, which is then opened by each of the other WNFE modules. All accounting and authentication interprocess communications are performed via this attach file as shown in Figure 2-18. The AAMs on different WNFEs establish TCP connections over which authentication and accounting information are exchanged.



81-53

Figure 2-18. Authentication and Accounting Data Paths

2.7.1.3 Configuration. Three configuration files are used by the AAM.

a. One file contains the dual homing information needed by the IP1822M.

b. The second file is used by the THPSAM and the VIPM. This file contains one entry for every terminal connected to the WNFE at the local site. For example, the entry for a terminal includes baud rate, location, authorized security level, mode, poll address, and line number.

c. The third file contains site-specific information that should not be dynamically modified. Examples are host name, host address, and time zone.

Each module requiring configuration data receives it from the AAM.

2.7.1.4 Authentication. The AAM maintains a single authentication file. This file contains information regarding users and processes. File records include such fields as name, permission, security, identification, group identification, password, user type, precedence, and transmission control code.

To process a connection request, the service modules (TCPSAM and THPSAM) need user, job, and network authorization information. The service module asks the AAM if the connection request is authorized. The AAM responds that it is either a valid or invalid connection request.

2.7.1.5 Logging. Each module in the WNFE, including the AAM, does some logging. Such logging can be of four types: connection status (audit) logging, module status logging, error logging, and debug logging.

Audit logging provides a means for determining the status of any connection being established on the WNFE. Audit logging also provides a log of all authentication requests and denials. Each module involved in processing a connection request logs pertinent information regarding the progress of the connection. The audit log provides a record of failed connections, completed connections, and connections in progress. The audit log also provides a record of connection requests and authentication status.

Module status logging records information regarding module resources such as buffers, lines or circuits, and disk space. At any time, the status log contains an accurate account of the resources being used by each module and the resources remaining for that module.

Error logging is done any time a module detects an error. Error log entries include appropriate information to help determine the reason for the error.

Debug logging provides a trace of module execution. At key places in the execution sequence for a module, an entry is made in the debug log to record that the module reached that point in the execution path.

Each module in the WNFE does debug, error, and module status logging. Any modules involved in processing a connection provides audit logging. These modules include TCPSAM, THPSAM, THPM, and TCPM. In addition, both OSAIM and AAM provide audit logging because they log authentication requests, modifications, and denials.

The various logs (i.e., the audit, module status, error, and debug logs) are combined in one physical log file. All log entries are passed to the AAM, which then writes the information in the log file.

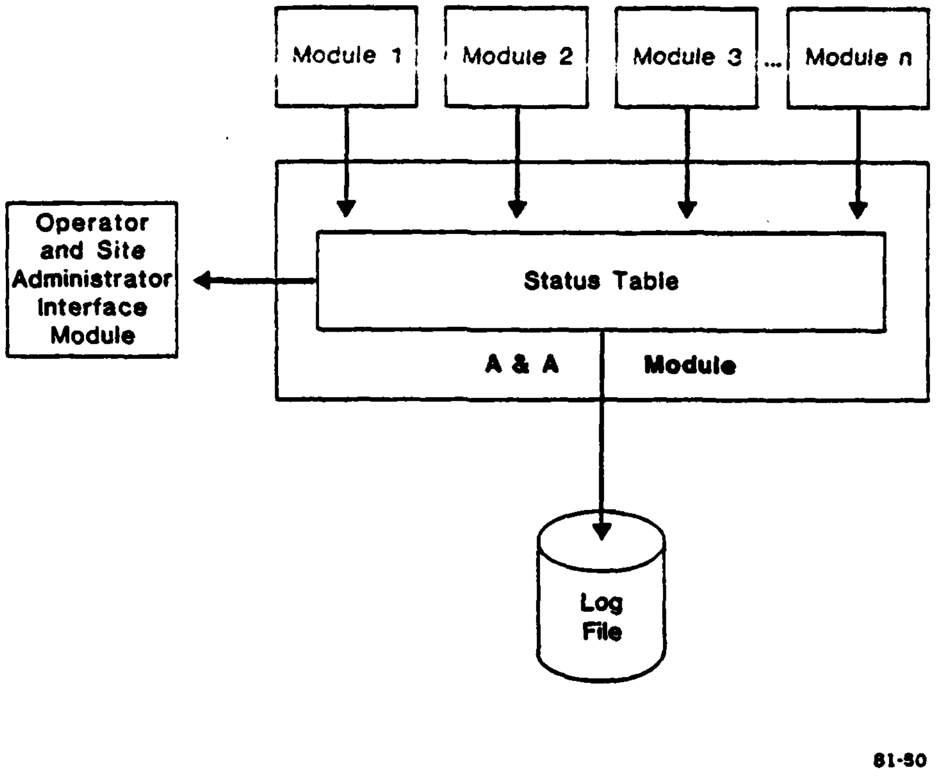


Figure 2-19. Status Data Paths

2.7.1.6 **Module Status Information.** The AAM provides status information to the Operator and Site Administrator Module (OSAIM) on request (see Figure 2-19). As the WNFE modules process a connection request, they transmit that connection's status to the AAM. This information is stored in a status table within the AAM. The status table contains current status information for all connections completed or in progress. In addition, some log information regarding authentication requests, modifications, and denials is recorded in the AAM status table.

A WNFE operator can monitor progress of connections at the site by requesting a connection status report through the OSAIM. The OSAIM obtains selected portions of the status table from the AAM and formats the data for the operator. The WNFE operator can also obtain the status of any WNFE module from the AAM through the OSAIM.

2.7.1.7 Configuration Control. The AAM coordinates activities of the WNFE protocol modules. The AAM also accepts updates to configuration and authentication information from the OSAIM. The data paths are shown in Figure 2-2Ø.

Instructions from the OSAIM to the AAM provide a way to modify, add, and delete information in the configuration and authentication files. Responses from the AAM inform the OSAIM when the request is completed.

When the OSAIM receives a request from a WNFE operator to modify, add, or delete configuration and authentication information, the OSAIM sends the AAM a request to modify a file. The AAM does the modification and logs the action in the audit file.

If a configuration modification is made, the AAM applies the modification to the appropriate configuration file and notifies the appropriate modules or device drivers to implement the modification. Thus, if the characteristics of a terminal are changed while the terminal is active, the AAM causes the changes to go into effect on the terminal as soon as the terminal becomes idle. This allows the system to be re-configured dynamically without rebooting the system.

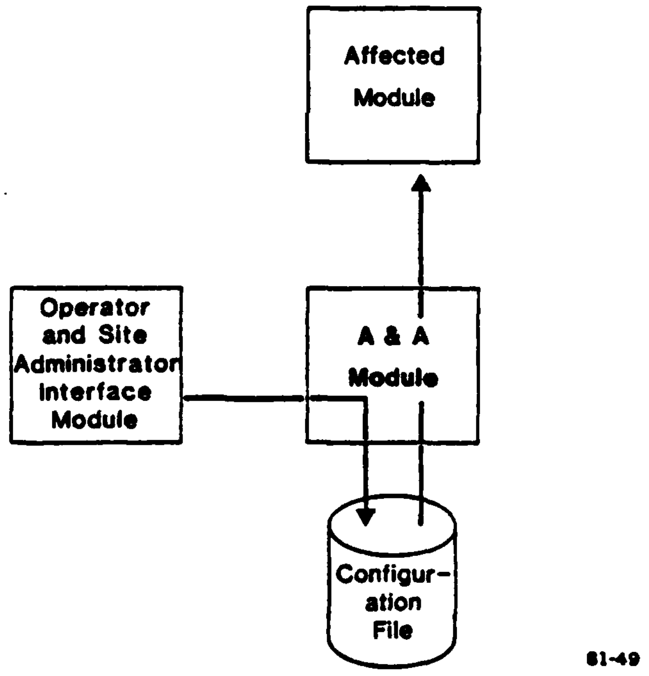
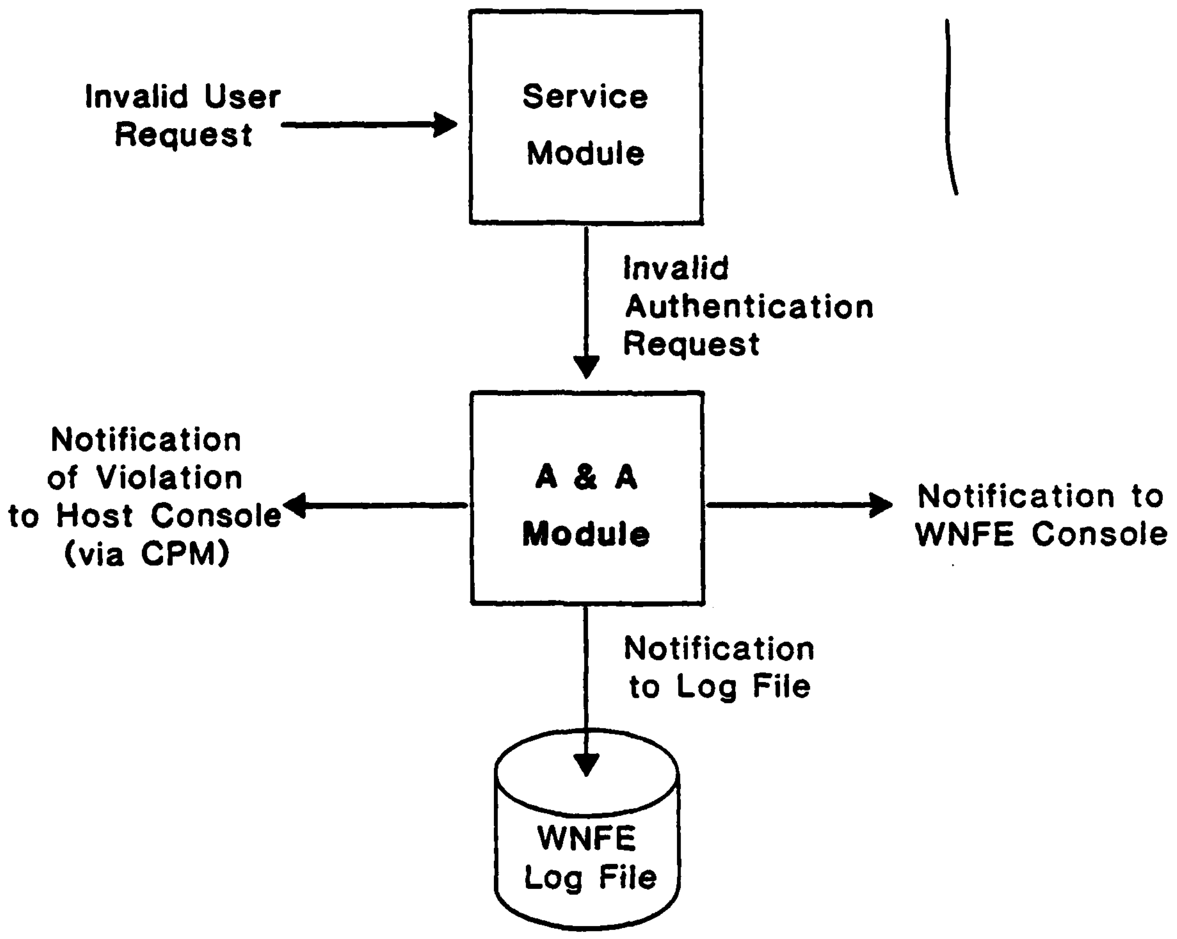


Figure 2-2Ø. Configuration Data Paths

2.7.1.8 Security Violation Reporting. If the AAM finds a connection request cannot be authorized, it logs the violation and sends copies of a violation message to both the host and WNFE consoles (see Figure 2-21).


Invalid User Request
Service Module
Invalid Authentication Request
Notification of Violation to Host Console (via CPM)
A & A Module
Notification to WNFE Console
Notification to Log File
WNFE Log File

81-166

Figure 2-21. Security Violation Reporting

2.7.1.9 WNFE to Host Log Transfer. The log information collected by the AAM must be transferred to the host for security and statistical inspection. This is done in the following way (see Figure 2-22). As the AAM writes each log record to the log file it marks those records that should be sent to the host. A submodule of the AAM, called the Host Format Converter, then reads these log records, converts those records that are of interest to the host to a format compatible with the host. The AA Service Access submodule then sends these records to the Host Accounting Module in the host. The Host Accounting Module writes these records to the Statistical Collection File.

2.7.1.10 Single Logon. The Single Logon feature is provided by extending the procedures used to authenticate local user requests for service. The required authentication data is passed between AAMs located in different systems via standard TCP connections.

2.7.1.11 Job Spawning. Job Spawning is provided by a mechanism similar to the procedures used to implement Single Logon. The authentication request from the local AAM to the remote AAM causes the remote AAM to send the spawn request to the remote host. The spawned job then requests a TCP LISTEN that matches the original user request.

2.7.2 Operator and Site Admin Interface Module. Authorized operators of the WNFE system include the Operator, the Site Administrator, the WWMCCS Site Coordinator (WSC), the WMMCCS ADP System Security Officer (WASSO), and any others granted the same permissions. All of these people are referred to here simply as operators.

The Operator and Site Administrator Interface Module (OSAIM) gives the operators control over the following:

a. Configuration of all terminals directly connected to the WNFE

b. Installation and monitoring of all WNFE access permissions for users at the site

c. Operation and limited maintenance of the PDP 11/70 computer

d. Monitoring of WNFE modules and their interface with the host machine and network

The OSAIM performs these functions in conjunction with the Authentication and Accounting Module (AAM).

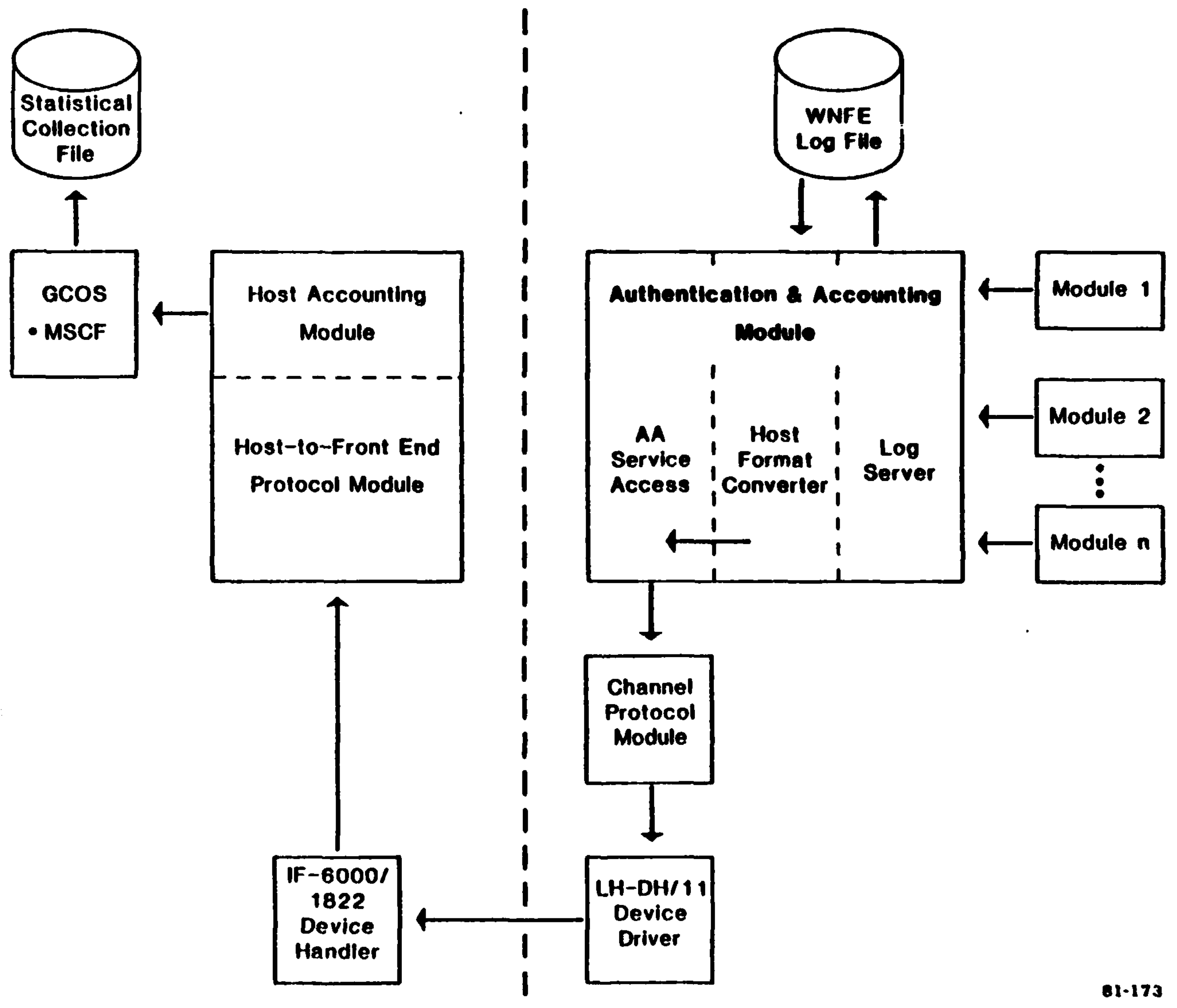


Figure 2-22. Log Transfer Data Paths

2.7.2.1 Interface Design. The Operator and Site Administrator Interface is completely menu-driven. A menu is a list of options appearing on the screen from which an operator must choose the next action he wishes to take. Any task that he wants to perform must be selected from a menu no actions other than the ones listed in the menus may be taken.

In many instances, the operator traverses several menus before reaching the menu listing the desired task. With each successive menu the options become more specifically defined.

Once the operator has selected the appropriate task from a menu, he then enters into a "conversation" with the interface. The OSAIM prompts for required information, and the operator responds to each prompt.

When the operator has responded to all of the prompts the OSAIM performs the requested task by calling UNIX routines or sending requests to the AAM. When the tasks is complete the operator is returned to the menu which started the scenario where he may select another task. The operator always has the option to leave the current menu and return to a previous menu or to start over with the main menu.

As shown in Figure 2-23, the Operator and Site Administrator Module (OSAIM) consists of a Menu Driver and an number of Task Modules. The operator interacts with the Menu Driver up to the point that a specific task is selected. Control then passes to the Task Module corresponding to the task selected. Each of the Task Modules communicates with the Authentication and Accounting Module (AAM) to authenticate the user's permission to perform the task and to perform the task.

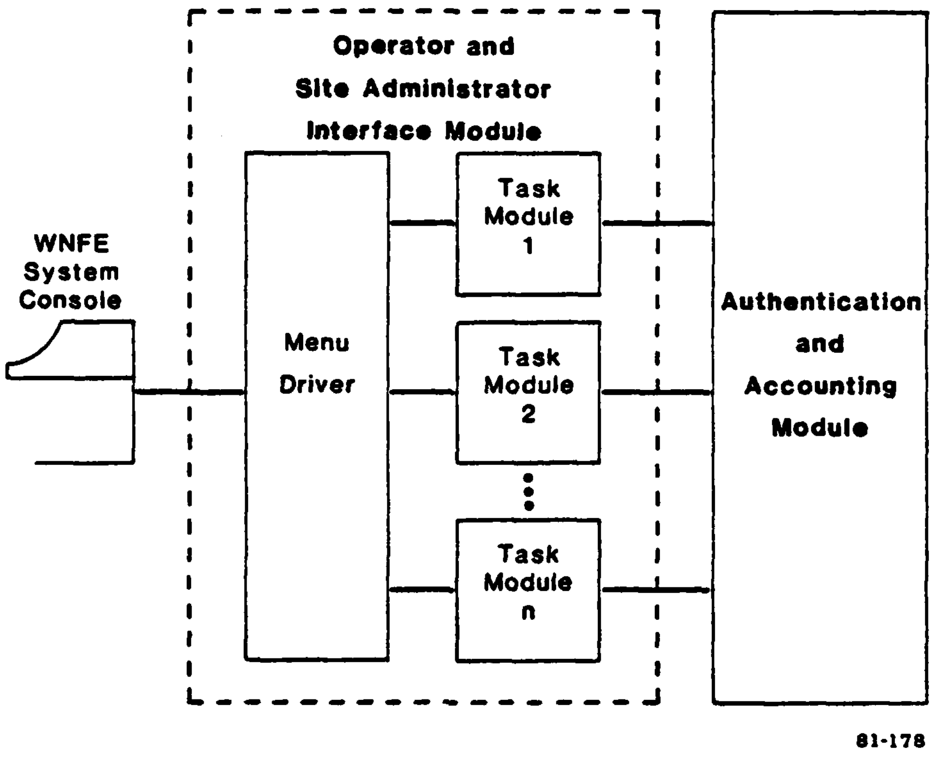


Figure 2-23. Operator and Site Administrator Interface

A permission matrix allows for the delegation of operator duties. Although the permission matrix is created with a default assignment of each task to an Operator, Site Administrator, or WASSO, permission for some tasks may be changed or reassigned. This allows flexibility in the assignment of tasks at each site.

2.7.2.2 **Operator Functions.** The functions of the operators of the WNFE fall into several categories. These include:

a. System operation and maintenance

b. Adding and maintaining users

c. Run-time monitoring

d. Configuring the system

e. Miscellaneous functions

These tasks are distributed among the different operators at a WNFE site. The OSAIM performs these tasks in a straightforward, reliable manner.

2.7.2.3 System Operation and Maintenance. The OSAIM provides options for

a. Powering-down the system

b. Removing or changing disk packs

c. Rebooting the system

d. Dumping a core image

e. Making a backup of the system pack

f. Mounting a file system

2.7.2.4 Adding and Maintaining User Information. The entries in the authentication file determine user access to the system and to the network. The OSAIM allows the operator to change the user information in this file by making requests to the AAM. The AAM then accesses the file. The following indicates the relations between user access and authentication file entries.

a. When a user is installed, an entry for the user is made.

b. When removing a user from the system, the user entry is removed from the file.

c. When changing a user's password, the new password is recorded in the file.

d. When disabling a user, a flag is set in the entry that makes the user unable to log in.

e. When enabling a previously disabled user, the flag is removed.

Finally, an operator can request that a report be printed that shows all authorized users of the WNFE and their status.

2.7.2.5 Run Time Monitoring. The OSAIM provides all essential functions for monitoring the running system. Monitoring options allow the operator to

a. Monitor the status of system resources

b. Monitor the status of running processes

c. Take appropriate action to control any users endangering the system

The operator can request several displays of the status of specific system resources. These include displays of

a. The state of each WNFE connection

b. The activity and user on each terminal accessing the WNFE

c. The buffer space being used by each of the WNFE modules

d. The status of all file systems on the WNFE

e. The resources being used by processes on the WNFE

The operator can get a display of all users currently using the system, or a separate display of running processes. He can modify the precedence of, and terminate selected processes.

If monitoring reveals unauthorized users or unauthorized activity, the operator can disable the user. This immediately logs the user off the system. The user cannot log in again until he has been re-enabled. The operator can also force a user to log off immediately without disabling his user code.

2.7.2.6 Configuring the System. The OSAIM provides functions for configuring the system. The first function is the creation of the file of terminal types. This file contains entries for each type of terminal connected to the WNFE. Each entry contains the hardware name of the terminal plus all characteristics of the terminal except the baud rate. (Terminal characteristics include parity, delays, half/full duplex, etc.).

The OSAIM provides options to modify terminal-type entries or to print a report showing all the current entries. Each terminal added to the system must be one of the types in this table. The system refers to the table when determining how to access each device.

A configuration contains one entry for each device connected directly to the WNFE. The operator can select task options that

a. Add a new device by adding an entry in the configuration file.

b. Remove a device by removing the appropriate entry.

c. Modify a device configuration by modifying the entry for the device.

d. Display the configuration of a device.

e. Display the entire contents of the configuration file.

2.7.2.7 **Miscellaneous Functions.** Among the options provided by the OSAIM which do not fall into any of the above categories are setting the date, changing the permission matrix, and sending messages to all users connected to the WNFE.

Some messages are sent to users automatically. For example, when the operator powers down the system the OSAIM automatically sends a warning message to all users.

APPENDIX C: CCITT Standard X.25

As packet switching proved itself in the network environment, interest in providing public data communications service using packet switching has grown. Networks providing public service are already in operation in the United States, Canada, and the United Kingdom. And, several are in advance stages of development in Japan, France, and several other countries. These networks are operated by government postal, telephone, and telegraph authorities, or by regulated private companies.

In an effort to provide standardized and compatible services throughout the world, these organizations cooperate through the International Telephone and Telegraph Consultative Committee (CCITT) to formulate recommendations for providing various types and levels of service. Several recommendations for packet switching operation have been adopted in recent years, most notably the X.25 recommendation specifying the protocol for subscriber access to public networks.

An important feature of X.25 is that it specifies only the interface between the customer's computer and his local connection point to the network. X.25 includes a protocol for managing the physical links between a customer and the network which is a version of the International Standards Organization (ISO) HDLC protocol, and a higher level virtual circuit protocol which provides for many simultaneous virtual circuits over the physical link. This latter packet level has many features of a transport protocol, but operates only between the subscriber computer on one side, and the local network switching node on the other side.

To achieve end-to-end service between two subscribers, the network must establish connections and transmit data between source and destination Data Communication Equipment (DCE's), using an internal protocol not specified in X.25. The destination node provides a third portion of a virtual circuit by using X.25 with the destination subscriber (see Figure C.1). Hence, end-to-end service is provided by a concatenation of at least three independent virtual circuits. The outer two links are governed by the X.25 protocol while the inner links are implemented as devised by each local network.

The X.25 level three protocol performs sequencing and flow control on a packet basis. Two logically separate full duplex data streams are provided in each connection; a normal and qualified stream, but both are sequenced and flow controlled together [Ref. 23].

Flow control uses a window mechanism with a size of from one to eight packets. The size is fixed at the time the connection is established, based on the class of service requested. No checksums or acknowledgements are used at level three, but the link level protocol provides common error detection and correction for all virtual circuits. An out of band signal, or interrupt, of eight bits is provided which must be acknowledged by the destination subscriber before a record can be transmitted.

Connections are established by a special call request packet, specifying the destination address and optional service features required. The request is forwarded through the network to the destination subscriber, who returns an "accept" or "reject" packet. Simultaneous requests by a pair of

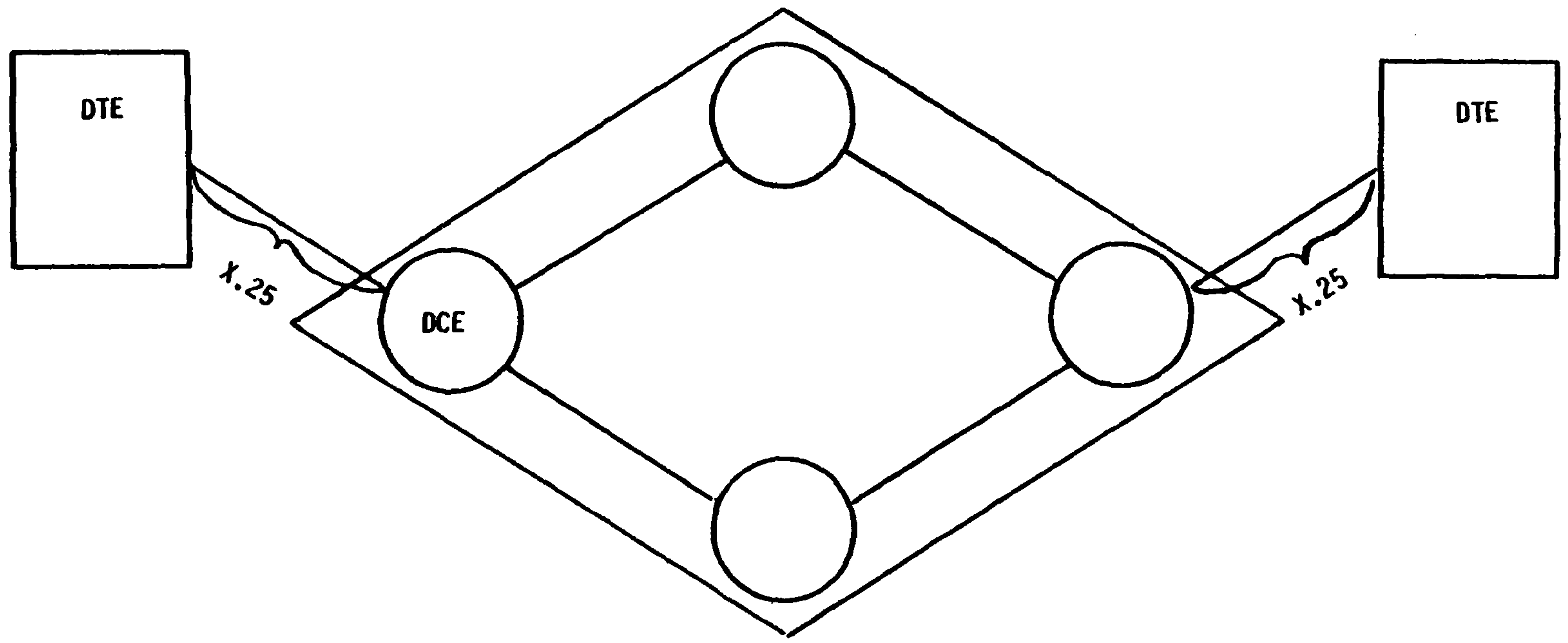


Figure C.1 X.25 Interface

Source: Kuo, F. F., "X.25 Interface," <u>Protocols and Techniques for Data Communication Networks</u>, Prentice Hall, Inc., p. 72, Figure 2-12.

subscribers to call each other may result in 0, 1 or 2 calls being set up depending on local network procedures. Connections may be cleared at any time, with any data in transit discarded. The user or network may also reset a virtual circuit, causing discard of any data in transit and a reset of sequence numbers and flow control, but not closing the connection.

Although X.25 specifies only the interface between customer and network it provides a basis for inferring the subscriber to subscriber service that will be offered by public networks. Some of the level three protocol features, such as interrupt and call establishment, are defined to have end-to-end significance. Other features, such as sequencing and flow control, are only required to have local significance, but may also be implemented to have end-to-end significance as in DATAPAC [Ref. 17]. Thus, when a receiving subscriber advances the flow control window, the effect on the sender's flow control window is uncertain and depends on local network implementation. Hence, the end-to-end service characteristics of public data networks require further definition beyond X.25 and will vary from network to network.

Because there is no end-to-end error control in X.25, users with high reliability requirements may need to implement their own, in an end-to-end protocol on top of X.25. The uncertainty of X.25 flow control may also required end-to-end flow control procedures. Some public network subscribers thus find themselves in need of end-to-end transport protocols on top of X.25 [Ref. 24]. To reduce this inefficiency, two approaches are possible. The transport protocols may be streamlined and adapted to take advantage of the high grade service offered by X.25 [Ref. 24, 25, 26]. Or, public network interfaces may be expanded to include a simpler datagram type service without connection establishment, sequencing or error control [Ref. 27, 28]. These functions could then be provided by an end-to-end transport protocol. In fact, during the original development of public packet networks, it was strongly argued that datagram and not virtual circuit service should be the basic service provided, with transport functions left to the users.

Finally, when public packet switching networks are interconnected, the uncertainties concerning end-to-end service characteristics are multiplied. A CCITT recommendation for interface between networks, X.75, has been formulated. X.75 closely resembles X.25 and specifies a virtual type protocol for the link between networks with a separate virtual circuit for each internet call. Although like X.25, X.75 addresses itself only to the link between networks and not to end-to-end architecture, it seems clear that the overall architecture will be a series of concatenated virtual circuits through each local network and across each internetwork link as shown in Figure C.2 [Ref. 29, 30]. This would increase the number of elements that must operate reliably and would increase the number of intermediaries participating in virtual circuit functions, raising doubts about the effectiveness of this approach.

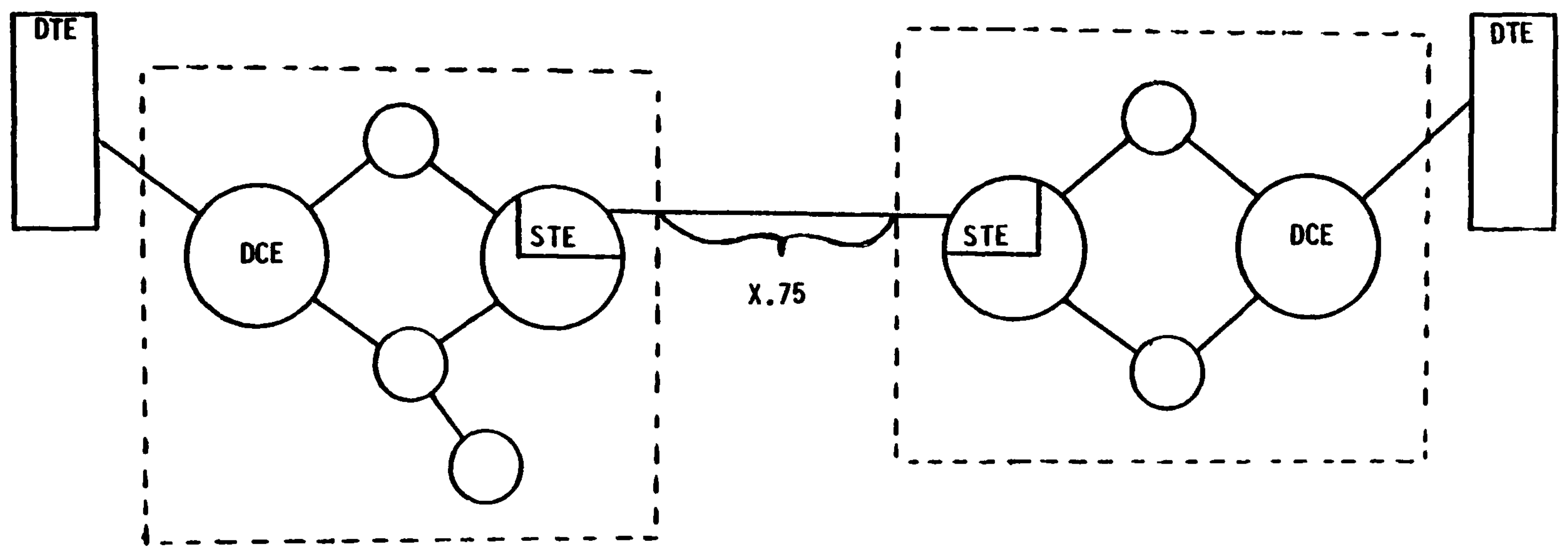


Figure C.2 X.75 Interface

Source: Kuo. F. F., "X.75 Interface," Protocols and Techniques for Data Communication Networks, Prentice Hall, Inc., p. 73, Figure 2-13.

LIST OF REFERENCES

1. Navy Fleet Material Support Office, Uniform Automated Data Processing System-Stock Points Design Documentation, Supply Point Logistics Integrated Communications Environment, Functional Description, 1 May 1980, FMSO Document No. F9420-001-9260, FD-SU01A.

2. Metcalfe, R. and Boggs, D., "Ethernet: Distributed Packet Switching for Local Computer Networks," Communications of the ACM, Vol. 19, No. 7, July 1976, pp. 395-404.

3. California University, Livermore, CA, Lawrence Livermore Laboratory, THC-A Simple High-Performance Local Network, by Knight J., and Itzkowitz, K., August 1980.

4. Gordon, R. L., Farr, W. W., and Levine, P., "Ringnet: A Packet Switched Local Network with Decentralized Control," Proceedings of the Local Area Network Symposium, May 1979, pp. 13-19.

5. Kahn, S. A., Stewart, R. L., Tolchin, S. G., and Healy, S. J., "Functional and Logical Description of a New Fiber-Optic Contention Bus Network," Proceedings of the IEEE 1980 COMPCON Fall, IEEE Society, pp. 269-272.

6. desJardins, R., and White, G. W., "ISO/ANSI Reference Model of Open Systems Interconnection," Proceedings Trends and Applications: Computer Network Protocols, IEEE Society, 1980, pp. 47-53.

7. Anderson, R. J., "Local Data Networks--Traditional Concepts and Methods," Proceedings of the Local Area Communication Network Symposium, Mitre Corporation and Institute for Computer Sciences and Technology, National Bureau of Standards, 1979, pp. 127-148.

8. University of Hawaii at Manoa, Department of Electrical Engineering, Honolulu, Hawaii, Design Issues for High Speed Local Network Protocols, by Kuo, F. F., May 1981.

9. Rawson, E. G., and Metcalfe, R. M., "Fibernet: Multimode Optical Fibers for Local Computer Networks," IEEE Transactions on Communications, July 1978, pp. 983-990.

10. Franta, W. R., and Chlamtac, I., Local Networks, Lexington, Massachusetts: Lexington Books, D. C. Heath & Co., 1981, p. 22.

11. Ibid., p. 23.

12. Luczak, E. C., "Global Bus Computer Communication Techniques," Proceedings of Computer Networking Symposium, IEEE Publishing Services, 1978, pp. 58-67.

13. Tannenbaum, A. S., Computer Networks, Englewood Cliffs, New Jersey: Prentice Hall, 1981, pp. 296-300.

14. Shoch, J. F., and Hupp, J. A., "Measured Performance of an Ethernet Local Network," Communications of the ACM, December 1980, Vol. 23, No. 12, pp. 711-721.

15. Davies, D. W., Barber, D. L. A., Price, W. L., and Solomondies, C. M., Computer Networks and Their Protocols, New York, New York: Wiley and Sons, 1981, pp. 125-131.

16. Tannenbaum, A. S., op. cit., pp. 373-374.

17. Sunshine, C. A., "Transport Protocols for Computer Networks," in Kuo, F. F., (ed.), Protocols and Techniques for Data Communication Networks, Englewood Cliffs, New Jersey: Prentice Hall, 1981, pp. 35-74.

18. University of Southern California, Information Sciences Institute, Transmission Control Protocol, Defense Advanced Research Projects Agency, Information Processing Techniques Office, Arlington, Virginia, August 1980, pp. 3-5.

19. Tannenbaum, A. S., op. cit., p. 368.

20. Sunshine, C. A., "Factors in Interprocess Communications Protocol Efficiency for Computer Networks," Proceedings of the National Computer Conference, IEEE Publishing Services, 1976, pp. 571-572.

21. Defense Communications Agency, WWMCCS Network Front End Development, WNFE Functional Description, by Allen, E. R., Draft Copy, November 1981.

22. U. S. Navy, Automated Data Processing Selection Office, Solicitation Document N66032-82-R-0007, Acquisition of Hardware, Software, and Services to Support the Supply Point Logistics Integrated Communication Environment (SPLICE), Project at 62 Navy Stock Point Sites, Request for Proposal, Draft Copy, 1 March 1982.

23. Tannenbaum, A. S., op. cit., pp. 237-245.

24. Eschenauer, E., and Bozenshi, V. O., "The Network Communication Manager: A Transport Station for the SGB Network," Proceedings of the Symposium on Computer Network Protocols, Institut de l'Ectricite, Universite de Liege au Sort Tilmen, Liege, Belgium, February 1978, pp. C2: 1-21.

25. Hertweck, F., Raubold, E., and Vogt, F., "X.25 Based Process to Process Communication," Proceedings of the Symposium on Computer Network Protocols, Institut de l'Ectricite, Universite de Liege au Sort Tilmen, Liege, Belgium, February 1978, pp. C3: 1-22.

26. Internetwork Working Group, "A Proposal for an Intranetwork End-to-End Protocol," INWG Note 96, July 1975, Proceedings of the Symposium on Computer Network Protocols, Institut de l'Ectricite, Universite de Liege au Sort Tilmen, Liege, Belgium, February 1978, pp. H: 5-25.

27. Jacquemant, Y. A., "Network Interprocess Communication in an X.25 Environment," Proceedings of the Symposium on Computer Network Protocols, Institut de l'Ectricite, Universite de Liege au Sort Tilmen, Liege, Belgium, February 1978, pp. C1: 1-6.

28. Pouzin, L., "Virtual Circuits vs. Datagrams--Technical and Political Problems," Proceedings of the AFIPS National Computer Conference, New York City, June 1976, pp. 483-494.

29. Grossman, G. R., Hinckley, A., and Sunshine, C. A., "Issues in International Public Data Networking," Computer Networks Journal, Vol. 3, No. 4, September 1979, pp. 259-266.

30. Sunshine, C. A., "Current Trends in Computer Interconnection," Advances in Data Communications Management, Philadelphis, Pennsylvania: Heyden and Sons, 1980, pp. 158-164.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Professor Norman F. Schneidewind, Code 54Ss Department of Computer Sciences Naval Postgraduate School Monterey, California 93940	1
4.	Lieutenant Jan Adams SMC 1942 Naval Postgraduate School Monterey, California 93940	1
5.	Lieutenant Commander Jerry Barnes SMC 2542 Naval Postgraduate School Monterey, California 93940	1
6.	Lieutenant Commander Kathleen Barret SMC 1087 Naval Postgraduate School Monterey, California 93940	1
7.	Lieutenant Sharon Crowder SMC 2518 Naval Postgraduate School Monterey, California 93940	1
8.	Captain Craig Opel SMC 1322 Naval Postgraduate School Monterey, California 93940	1
9.	Lieutenant Ricard Arana Courrejolles Ministerio de Marina Central de Procesamiento de Datos de la Marina de Guerra Av. Salaverry S/N Lima, Peru, South America	1

		No. Copies
10.	Captain Kenneth A. Inman, Jr. Computer Science School Marine Corps Development and Education Command Quantico, Virginia 22134	2
11.	Lieutenant Robert C. Marthouse SMC 1493 Naval Postgraduate School Monterey, California 93940	2
12.	Lieutenant Colonel J. Mullane Marine Corps Representative Code 0309 Naval Postgraduate School Monterey, California 93940	1